НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет
Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:
Завідувач кафедри
_____ Олександр КОВАЛЬ
«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні технології моніторингу довкілля»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям»

Виконала:
студентка IV курсу, групи ТМ-62
Семіон Вікторія Ігорівна _____

Керівник:
старший викладач
Бандурка Олена Іванівна _____

Рецензент:
доцент каф. ТЕУ та АЕС
Сірий Олександр Анатолійович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Інформаційні технології моніторингу довкілля

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр КОВАЛЬ
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту
Семіон Вікторії Ігорівні
(прізвище, ім’я, по батькові)

1. Тема роботи «Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям»
керівник роботи Бандурка Олена Іванівна, старший викладач
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи 3 жовтня 2019 р.
3. Вихідні дані до роботи мова програмування JavaScript, середовище Android Studio, фреймворк React Native, бібліотека Jest

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) проаналізувати проблематику серцево-судинних захворювань, розглянути їх типи та електрокардіографію як метод діагностування серцево-судинних захворювань, розробити та протестувати програмне забезпечення для отримання ЕКГ сигналів та пульсу мобільним пристроям

5. Перелік ілюстративного матеріалу: мета дипломної роботи, актуальність роботи, електрична активність серця, електрокардіограма, конкуренти, засоби розробки, діаграма прецедентів, програмна реалізація, тестування додатку, системні вимоги додатку, робота з додатком, відображення результатів, порівняння додатку з конкурентами, висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”11” жовтня 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	25.10.19	
2.	Вивчення та аналіз задачі	03.02-11.02.20	
3.	Розробка архітектури та загальної структури системи	12.02-20.02.20	
4.	Розробка структур окремих підсистем	21.02-28.02.20	
5.	Програмна реалізація системи	02.03-11.03.20	
6.	Оформлення пояснювальної записки	12.03-29.05.20	
7.	Захист програмного продукту	05.06.20	
8.	Передзахист	09.06.20	
9.	Захист	18.06.20	

Студент _____ Вікторія СЕМІОН
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Олена БАНДУРКА
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Дипломну роботу виконано на 62 аркушах, вона містить 4 додатки та перелік посилань на використані джерела з 21 найменування. У роботі наведено 27 рисунків.

Метою даної дипломної роботи є розробка автоматизованої системи для передачі ЕКГ сигналів портативним мобільним пристроям. Створений програмний продукт розроблено за допомогою фреймворку React Native для забезпечення доступу мобільним пристроям. Тим більш, він повинен мати інтуїтивно зрозумілий та інтерактивний інтерфейс користувача.

Ключові слова: ЕКГ, система передачі, мобільні пристрої, автоматизована система, серцево-судинні захворювання.

ABSTRACT

The thesis is presented in 62 pages. It contains 4 appendixes and bibliography of 21 references. There are 27 images in the thesis.

The purpose of this thesis is developing an automated system for receiving ECG signals to portative mobile devices. The developed software product should be created with framework React Native to provide access to mobile devices. Furthermore, it must have an intuitive and interactive user interface.

Keywords: ECG, transmission system, mobile devices, automated system, cardiovascular diseases.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 6
ВСТУП 7
1 ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПЕРЕДАЧІ ЕКГ СИГНАЛІВ ПОРТАТИВНИМ МОБІЛЬНИМ ПРИСТРОЯМ 9
2 АНАЛІЗ ПРОБЛЕМИ ДІАГНОСТИКИ СЕРЦЕВО-СУДИННИХ ЗАХВОРЮВАНЬ ЗА ДОПОМОГОЮ ЕКГ 11
2.1 Проблема розповсюдженості серцево-судинних захворювань 11
2.2 Типи серцево-судинних захворювань та методи їх діагностики 12
2.3 Електрокардіографічний метод діагностики 15
2.4 Системи передачі ЕКГ сигналів портативним мобільним пристроям 19
2.5 Висновки до розділу 23
3 ЗАСОБИ РОЗРОБКИ 25
3.1 Середовище розробки Android Studio 25
3.2 Технологія Bluetooth Low Energy 27
3.3 Мова програмування JavaScript 30
3.4 Мова гіпертекстової розмітки HTML 34
3.5 Таблиця каскадних стилів CSS 36
3.6 Висновки до розділу 37
4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ 38
4.1 Структура програмного забезпечення 38
4.2 Висновки до розділу 48
5 РОБОТА КОРИСТУВАЧА З СИСТЕМОЮ 49
5.1 Системні вимоги 49
5.2 Робота користувача з програмним продуктом 50
5.3 Висновки до розділу 58
ВИСНОВКИ 59
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 61
Додаток 1 63
Додаток 2 74
Додаток 3 65
Додаток 4 82

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ССЗ — серцево-судинні захворювання;

ЕКГ — електрокардіограма;

BLE (англ. Bluetooth Low Energy) — бездротова технологія Bluetooth з низьким енергоспоживанням;

GAP (англ. Generic Access Profile) — протокол загального доступу, що означає сумісність приладу з обладнанням інших виробників, які мають той же стандарт;

GATT (англ. General Attribute Protocol) — профіль загальних атрибутів, що містить загальні специфікації відправки та прийому невеликих порцій даних;

HTML (англ. Hypertext Markup Language) — стандартизована мова гіпертекстової розмітки, що використовується для створення веб-сторінок і веб-додатків;

CSS (англ. Cascading Style Sheets) — формальна мова опису зовнішнього вигляди документу, що є написаним на мові розмітки ;

DOM (англ. Document Object Model) — інтерфейс, що дозволяє програмам та скриптам отримати доступ до документів, змінювати оформлення, структуру;

SVG (англ. Scalable Vector Graphics) — мова розмітки масштабованої векторної графіки;

D3 (англ. Data-Driven Documents) — бібліотека мови програмування JavaScript для маніпулювання документами на основі даних;

W3C (англ. World Wide Web Consortium) — організація що розробляє та впроваджує технологічні стандарти;

ES (англ. ECMAScript) — мова програмування яка є основою для інших скриптових мов;

JSX (англ. JavaScript eXtensible Markup Language) — синтаксис, який робить можливим використання HTML/XML подібного тексту разом з JavaScript.

ВСТУП

Захворювання серцево-судинної системи – це тип захворювань, вплив якого важко переоцінити оскільки ССЗ продовжує знижувати якість та тривалість життя кожної людини у світі протягом останніх десятиліть.

Проблема збільшення кількості серцево-судинних захворювань серед громадян нашої країни є досить суттєвою. На жаль, темпи смертності в Україні відповідають рівню слаборозвинених країн, що пояснює той факт, що Україна займає 150-е місце серед 223 країн за середньою тривалістю життя. Заради розуміння масштабності проблеми слід зазначити, що 30% чоловіків та 15% жінок не доживають до пенсійного віку [1]. Серцево-судинні захворювання відіграють велику роль у цій проблемі оскільки Україна продовжує займати найвищі позиції за смертністю від цього типу захворювань серед чоловіків та жінок.

Таким чином, наразі спостерігається значна різниця між станом здоров’я середньостатистичного громадянина Західної (у тому числі й України) та Східної Європи і якщо, наприклад, у 1991 році показник захворюваності ССЗ в України на 30% перевищував відповідний середньоєвропейський рівень, то станом на останні роки рівень захворюваності в країнах Східної Європи вже у 2 рази менше при порівнянні з Україною [2]. Саме тому спостерігання за здоров’ям серцево-судинної системи є пріоритетною задачею задля зменшення рівня смертності в нашій країні.

Вчасне діагностування разом з правильним лікуванням є важливим кроком на шляху до покращення якості та продовження тривалості життя при усіх типах захворювань і ССЗ не є винятком. Одним з основних методів, що використовується для діагностики у кардіології, є електрокардіографія.

ЕКГ – це зображення електричної активності серця, що має вигляд кривої. Таким чином, електрокардіографія зображує електричні імпульсі, що

формуються у спеціальній групі клітин у передсердях та змушують відділи серця скорочуватися. Саме ЕКГ дозволяє провести діагностику таких частих захворювань, як ішемічна хвороба серця, неправильне проведення імпульсів або їх блокада та порушення ритму серцебиття (наприклад, тахікардія). Цей тип діагностування має певні переваги – ЕКГ є однією з найбільш доступних та безпечних методів, може використовуватися для обстеження пацієнтів у важкому стані, вагітних жінок та дітей. Таким чином, ЕКГ не має ніяких протипоказань та використовується не тільки у медичних установах, а також в аптеках, тренажерних залах та навіть у домашніх умовах.

Основною метою цієї роботи є розроблення автоматизованої системи, що дозволяє користувачам спостерігати за їх ЕКГ сигналами. Актуальність проекту полягає у відсутності безкоштовних аналогів, які можуть бути використовувані з цією ціллю та у відсутності будь-яких аналогів на українській мові. Розроблений програмний продукт надає можливість громадянам країни слідкувати за здоров'ям серцево-судинної системи та зробити цей процес зручнішим та можливим у домашніх умовах.

Дипломна робота містить 5 розділів. Перший розділ містить інформацію про постановку задачі автоматизованої системи передачі ЕКГ сигналів портативним мобільним пристроям.

Другий розділ описує проблему розповсюдженості серцево-судинних захворювань, їх типи та види діагностики, а також: створення та розвинення електрокардіографії як виду діагностики та сучасні системи, що дозволяють отримати електрокардіограму за допомогою мобільного додатку.

Що стосується третього розділу, він описує усі засоби розробки, які були використанні у ході створення потрібного програмного продукту.

Четвертий розділ містить опис програмної реалізації: структура розробленої системи, опис основних компонентів.

П'ятий розділ детально описує процес взаємодії користувача з програмним продуктом та містить інформацію про потрібні для цього системні вимоги.

1 ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПЕРЕДАЧІ ЕКГ СИГНАЛІВ ПОРТАТИВНИМ МОБІЛЬНИМ ПРИСТРОЯМ

Метою даної дипломної роботи є розроблення програмного продукту, що надає змогу отримати зображення ЕКГ сигналів, спираючись на які, користувач зможе зробити припущення стосовно стану здоров’я його серцево-судинної системи, отримати інформацію відносно норми результатів ЕКГ та пульсу, перевірити свій пульс та поділитися результатами ЕКГ з лікарем. Потенційними користувачами можуть виступати люди будь-якого віку, оскільки проблема ССЗ наразі зачіпає не тільки людей похилого віку, а й дітей та підлітків оскільки генетика та неправильний спосіб життя суттєво впливають на ризик розвитку серцево-судинних захворювань. Тим більше, програмний продукт буде цікавий людям будь-якого соціального статусу, оскільки для його успішного використання достатньо мати смартфон Android та будь-який прилад пригідний для отримання ЕКГ сигналів та пульсу. Наразі існує велика кількість відповідних приладів зовсім різної цінової політики. Сама система є безкоштовною і не потрібує ніяких додаткових витрат. Таким чином, програмний продукт повинен мати наступний функціонал:

– підключення до потрібного приладу задля отримання потрібної для виміру ЕКГ та пульсу інформації;
– відображення електричної активності серця у вигляді кривої (ЕКГ);
– вимір пульсу користувача;
– відображення інформації стосовно норми ЕКГ;
– відображення інформації стосовно норми пульсу та визначення занадто низького і високого показника;
– можливість відправити електрокардіограму.

У ході створення зазначеної автоматизованої системи використовувалася

мова програмування JavaScript задля забезпечення інтерактивності та розробки алгоритму.

Інтерфейсу користувача був розроблений за допомогою мови розмітки гіпертексту HTML для наповнення програмного продукту потрібною інформацією та таблиці каскадних стилів CSS для поліпшення візуальної складової.

Оптимізації роботи з динамічним відображенням даних забезпечується завдяки компонентам бібліотеки React Native. Підключення до потрібного приладу здійснюється за допомогою технології Bluetooth. Обмін даними був розроблений завдяки протоколу GATT. З ціллю підвищення оптимізації та прискорення процесу розробки використовувався власноруч модернізований SDK, створений на базі Polar SDK, який у свою чергу написаний на популярній мові програмування Java.

2 АНАЛІЗ ПРОБЛЕМИ ДІАГНОСТИКИ СЕРЦЕВО-СУДИННИХ ЗАХВОРЮВАНЬ ЗА ДОПОМОГОЮ ЕКГ

Питання створення зручного та доступного більшості громадян методу діагностики серцево-судинних захворювань або програмного забезпечення, що дозволить перевіряти стан свого здоров’я задля своєчасного звертання до лікаря наразі є досить актуальним, оскільки швидкість розповсюдження цього типу захворювань у всьому світі і у пострадянських країнах шокуюча і вона зростає з кожним роком. Причин для цього декілька: недостатньо активний спосіб життя, зловживання продуктами з підвищеною кількістю транс-жирів, зловживання алкоголем тощо.

2.1 Проблема розповсюдженості серцево-судинних захворювань

Відповідно до Всесвітньої Організації Здоров’я, незважаючи на поліпшення медичного обладнання, відкриття нових технологій лікування та діагностики, поступове зниження кількості курців і так далі, розповсюдженість та смертність через цей тип захворювання буде продовжувати збільшуватися – якщо на момент 2010 року щорічно помирало 18.1 мільйонів людей, то станом на 2020 рік приблизна кількість померлих від ССЗ становить вже 20.5 мільйонів людей у всьому світі. Завдяки відповідному прогнозу на 2030 рік стає зрозумілим, що кількість жертв ССЗ приблизно буде становити вже 24,2 мільйони. Що стосується відсотку смертей від серцево-судинних захворювань відповідно до усіх смертей, вже станом на 2010 рік він становив 30.8%, а прогнози на 2020 та 2030 рік становлять вже 31.5 та 32.5 відсотків відповідно [3].

Для оцінювання проблеми розповсюдженості серцево-судинних захворювань та їх впливу на популяцію надзвичайно важливим є показник DALY. Цей показник надає інформацію про кількість втрачених років з поправкою на інвалідність та може вважатися як один втрачений рік «здорового» життя. Рівень цього показника серед населення використовується для оцінки розриву між

поточним станом здоров'я громадян та їх ідеальним здоров'ям. Ідеальне здоров'я громадян відповідає ситуації, коли всі люди доживають до похилого віку, не мають інвалідності та хвороб. DALY знаходиться як сума років життя втрачених через передчасну смерть (YLL) та років, що втрачені через втрату працездатності (YLD):

$$DALY = YLL + YLD, \tag{2.1}$$

Сам показник YLL є добутком кількості смертей (N) на очікувану тривалість життя у розглянутому віці (L):

$$YLL = N \times L, \tag{2.2}$$

Що стосується показника YLD, який також необхідний для підрахування DALY, для його оцінки за конкретною причиною у конкретний часовий період треба знайти добуток кількості інцидентів у відповідний період (I), середньої тривалості захворювання (DW) та вагового коефіцієнту (L), що демонструє тяжкість захворювання від 0 (абсолютно здоровий) до 1 (мертвий) [4]:

$$YLD = I \times DW \times L, \tag{2.3}$$

Показник DALY станом на 2010 рік становив 153 мільйони у всьому світі, а прогнози на 2020 та 2030 рік становлять 169 та 187 мільйонів відповідно. Відповідно до ВОЗ, DALY від серцево-судинних захворювань становить 10.4% від усіх DALY у світі, а показники на наступні два десятиліття мають значення 11% та 11.6% [3]. Оцінюючи DALY у Європі та Центральній Азії, можна відмітити що саме в Україні, Білорусії, Болгарії та Росії знаходяться найвищі показники. Таким чином, найбільший вплив на кількість років з поправкою на інвалідність в Україні мають саме серцево-судинні захворювання, а саме ішемічна хвороба серця та інсульт [5].

2.2 Типи серцево-судинних захворювань та методи їх діагностики

Зазвичай усі серцево-судинні захворювання поділяють на сім типів:

– поразка судин серця (наприклад, атеросклероз, який є хронічним захворюванням що характеризується відкладанням ліпідів та солей кальцію на внутрішніх стінках сосудів що призводить до їх ущільнення та зменшенню

просвіту);

– вади клапана серця (порушення роботи серця через дефекти у клапанному апарату, які можуть буть вродженими через поразку плода або генетичних порушень чи набутими під час життя) ;

– запальні захворювання серця (можуть виникнути у різних частинах серця – у серцевому м’язі, що має назву міокард або всередині внутрішньої чи сполучної оболонки серця що мають назву ендокард та перикард відповідно);

– ішемічні ураження (характеризуються зменшенням або повним припиненням крові до міокарда та призведе у першому випадку до ішемічної хвороби та до інфаркту міокарда у другому);

– патологічні зміни (являються невідворотними дефектами у діяльності серця – наприклад, серцева недостатність що характеризується неможливістю міокарда створити енергію необхідну для прокачування потрібної кількості крові по організму).

– артеріальна гіпертензія (при цьому захворюванні спостерігається підвищення тиску крові в артеріях великого кола кровообігу)

– порушення ритму та провідності (наприклад, аритмія серця що характеризується неправильним функціонуванням електричних імпульсів що викликають серцеве скорочення та призводить до неритмічного биття серця) [6].

Серед усіх серцево-судинних захворювань можна виділити найбільш розповсюджені. Ситуація у всьому світі стосовно найчастіших ССЗ однакова. Таким чином, відповідний список має таких вигляд (хвороби розташовані за убуванням кількості хворих):

– інфаркт міокарда (характеризується недостатністю кількості кисню для роботи серця через зменшення або припинення постачання крові, що є носієм кисню та може виникнути через вже зазначений вище атеросклероз або накопичення нальоту в артеріях, який може бути холестерином, жиром або іншими речовинами);

– інсульт (ця хвороба виникає через проблеми постачання крові до мозку, що виникає як результат закупорки сосудів зазвичай тромбом при ішемічному

інсульті, розірванні стінок кровоносних судин при геморагічному інсульті або крововилив між м’якою та павутинною мозковими оболонками при нетравматичному субарахноїдальному крововиливі та призводить до пошкодження мозку [7]);

– серцева недостатність (характеризується таким постачанням крові до серця, що не відповідає нормі тобто серце продовжує качати кров, але робить це з недостатньо швидкістю що негативно впливає на функціонування організму) ;

– аритмія (призводить до неефективної роботи серця через неправильний ритм, що охоплює велику кількість відхилень – наприклад, занадто швидкий або занадто малий ритм);

– вади серця (наприклад, неправильне закриття клапанів, що дає крові змогу просочитися та має назву регургітація або стеноз, який являє собою порушення в роботі клапанів та характеризується невідкриттям клапанів в серці, що не дає змогу крові протікати) [8].

На разі існує багато видів діагностики ССЗ завдяки розвиненим технологіям. Однією з найважливіших типів діагностик є лабораторне дослідження, оскільки воно використовується завжди та дає змогу оцінити стан здоров’я пацієнта в цілому. При цьому дослідженні увага звертається на кількість глюкози, електролітів, холестерину тощо. Тим більше, особливе значення для діагностування серцево-судинних грають такі показники, як: кількість ліпопротеїдів, креатину, ферментів і так далі. Звісно ж, важливу роль при оцінюванні стану здоров’я серцево-судинної системи має також вимір артеріального тиску, яке кожна людина може здійснити вдома при наявності тонометра, в аптеці і, безумовно, у будь-якій медичній установі. Досить часто використовується рентгенограма грудної клітини, через її ефективність у діагностуванні симптомів ішемічної хвороби серця – наприклад, його збільшення. Задля оцінки коронарних та інших артерій використовуються ангіографічні дослідження, що дозволяють виявити звуження судин, їх збільшення або порушення прохідності (атеросклеротичною бляшкою або тромбом). Ангіографічна діагностика виконується за допомогою речовини, що вводиться у

судини за допомогою катетеру та рентгенограми. Таким чином, ангіографія потребує попереднього огляду та дозволу анестезіолога на цю процедуру. Апарат УЗІ також використовується задля діагностики серцево-судинних захворювань. Прикладом може послугувати ехокардіографія, що вимірює кількість викиду крові, стан стінок коронарних судин у покої чи при фізичних навантаженнях. Тим не менш, електрокардіографія, що відбувається у медичних установах на кардіографі та є фіксуванням електричних коливань під час роботи серця, є важливою для діагностування ССЗ. Таким чином, завдяки результатам ЕКГ досліджень, кардіолог має змогу правильно поставити діагноз, опираючись на відхилення основних зубців. На відміну від ангіографічного дослідження та ехокардіографії, електрокардіографія не має ніяких протипоказань та може бути зроблена на будь-якій людині. Досить відомим є зараз холтерівське дослідження, що бере за основу електрокардіографію. Для цього типу діагностики використовується прилад холтер, що є типом електрокардіографічного портативного приладу та здійснює моніторинг серця не менш ніж 24 годин [9]. На відміну від ЕКГ, при якій вимір здійснюється тільки протягом 5-10 секунд, холтерівське дослідження являє собою постійні виміри показників ЕКГ на відповідному часовому протязі не менше однієї доби. Таким чином, цей тип діагностики є більш точнішим та краще надає інформацію про здоров’я серцево-судинної системи. Тим не менш, цей тип дослідження є незручним оскільки прилад має досить великий розмір та високу вартість. Отже, найзручнішою для пацієнтів можна визначити діагностику за допомогою ЕКГ через відсутність протипоказань, швидкість проведення сеансу спостереження та порівняно низьку ціну.

2.3 Електрокардіографічний метод діагностики

Історію електрокардіографії можна поділити на три етапи: до Вілляма Ейнтговена, епоха Ейнтговена та після Ейнтговена, оскільки саме він зробив найбільший внесок у розробку цього типу дослідження. Ще у 1887 році на конгресі фізіологів у Лондоні він продемонстрував криву, що демонструвала

потенціали дій серця і була зроблена за допомогою струнного гальванометру та вже через два роки назвав її «кардіограма». Ще через п’ять років він найменував усі зубці розробленої їм кардіограми P, Q, R, S, T, а потім – U. Таким чином, вже у 1901 році видатний фізіолог розробив перший у світі кардіограф та продовжував розвиток цієї методики, за що й отримав Нобелівську премію у 1924 році. Американські фізіологи Вільсон та Гольдбергер теж зробили відчутний внесок, завдяки яким у 1952 році Всесвітня Організація Здоров’я запровадила протокол запису та розшифровки ЕКГ [10].

Отже, електрокардіографія використовується вже майже 70 років та стає все більш і більш популярною. Ще 15 років тому вважалося, що розвиток цього типу дослідження досяг свого максимуму. Тим не менш, протягом останніх 10 років відбулося декілька інноваційних відкриттів, які умовно можна поділити на три групи: бальні системи класифікації ЕКГ, нові цифрові методи діагностики та портативні електрокардіографічні прилади [11]. Що стосується бальних систем класифікації ЕКГ, вони оцінюють різноманітні амплітудно-часові параметри електрокардіограми та призначають кожному з таких параметрів кількість балів, що залежить від ступеню відповідності визначеній нормі. Перш за все, цей тип ЕКГ використовуються для визначення ступеня ураження серцево-судинної системи пацієнта від інфаркту міокарда. Тим не менш, відповідно до останніх досліджень, бальні системи також використовуються заради прогнозування ступеня можливості ССЗ у людей, які не мали інфаркт міокарда [12]. Що стосується нових методів ЕКГ-діагностики, вони базуються на використанні сучасних підходів до математичного опису та використанні складніших параметрів та характеристик для аналізу електрокардіограми при порівнянні зі звичними амплітудно-часовими показниками, що робить можливим діагностування патологічних змін на ранніх стадіях. Портативні електрокардіографічні прилади стають все більш популярними, оскільки є частиною досить широкої тенденції POST (point-of-care testing), що дає можливість проводити медичний тест пацієнтом самостійно у зручному для нього місці і не потребує втручання лікаря. Більшість з розроблених на даний момент

портативних електрокардіографічних приладів на даний момент мають можливість реєструвати тільки одне електрокардіографічне відведення, тобто є одноканальними. Не дивлячись на те, що такими типами ЕКГ приладів, звісно ж, неможливо досягти максимальної точності, більшість людей вважають їх надзвичайно важливими та корисними. Так, наприклад, відомий винахідник Норман Холтер вважав, що не можна довіряти виключно показникам ЕКГ, які були отримані у медичному центрі протягом усього 5-10 хвилин, оскільки не можна бути впевненими, що результат достовірний. Серце скорочується 12000 разів у добу, а на плівці записуються лише 10-12 комплексів [13]. Таким чином, портативні пристрої, що дозволяють провести ЕКГ пацієнту самостійно, широко використовуються у всьому світі, оскільки дають змогу часто проводити електрокардіографію, що безумовно допоможе слідкувати за своїм здоров'ям та надати лікарю інформацію про відповідні спостереження під час огляду.

Задля розуміння роботи електрокардіографічного методу діагностики необхідне знання структури серця. Функція серця полягає у перекачуванні крові до всіх частин тіла для забезпечення органів киснем. Серце можна розділити на чотири частини: два передсердя (праве та ліве) та два ніжні шлуночки (правий та лівий). Передсердя отримують кров, що надходить з тканин або легенів через венозну систему, а шлуночки викидають кров назустріч тканинам або легеням. У серці відбувається електрична активність, яка пов'язана з деполяризацією (зменшення потенціалу спокою) та реполяризацією (повернення потенціалу до рівня, який був перед деполяризацією), що супроводжують приплив крові. Імпульс спочатку починається у синоатріальному вузлі та протікає спочатку через передсердя, досягаючи до атріовентикулярного вузла та генеруючи скорочення передсердь. Потім відбувається невелика пауза та струм починає протікати до шлуночків, генеруючи їх скорочення. У кінці струм надходить до волокон Пуркіньє та відбувається реполяризація серцевої тканини.

Таким чином, електрична активність серця, що зображена на рисунку 2.1, має наступні етапи [14]:

1. Передсердя починають деполяризуватися.

2. Передсердя деполяризуються.
3. Передсердя реполяризуються, а шлуночки починають деполяризуватися на верхівці.
4. Шлуночки деполяризуються.
5. Шлуночки починають реполяризуватися на верхівці.
6. Шлуночки реполяризуються.

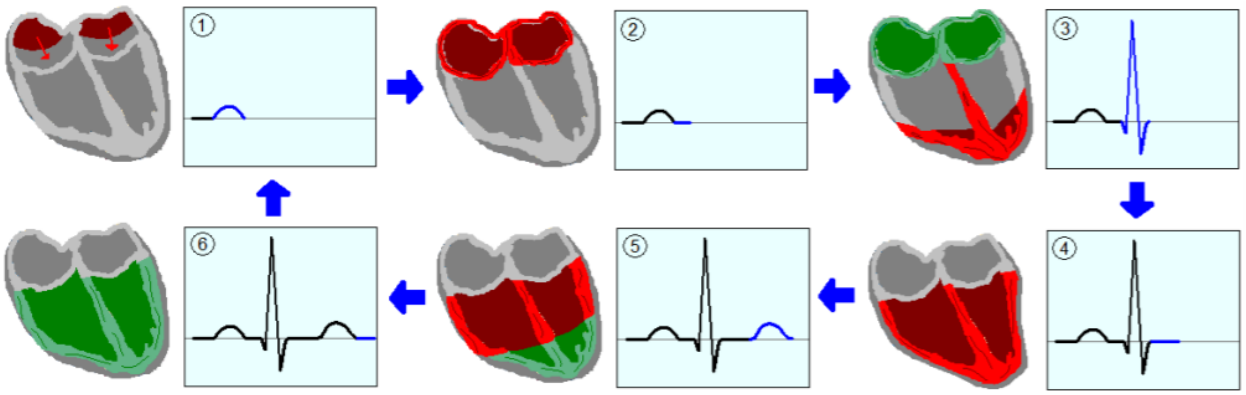


Рисунок 2.1 — Електрична активність у серці

Електрокардіограма, частина якої зображена на рисунку 2.2, містить різні комплекси, що позначаються за допомогою літер P, Q, R, S, T. Комплекс QRS є найбільшою та центральною частиною електрокардіограми. Комплекс QRS та хвиля T зображують активність шлуночків, а хвиля P є зображенням активності передсердь. Комплекс QRS має більшу амплітуду при порівнянні з хвилею P оскільки шлуночки мають більше м’язів ніж передсердя.

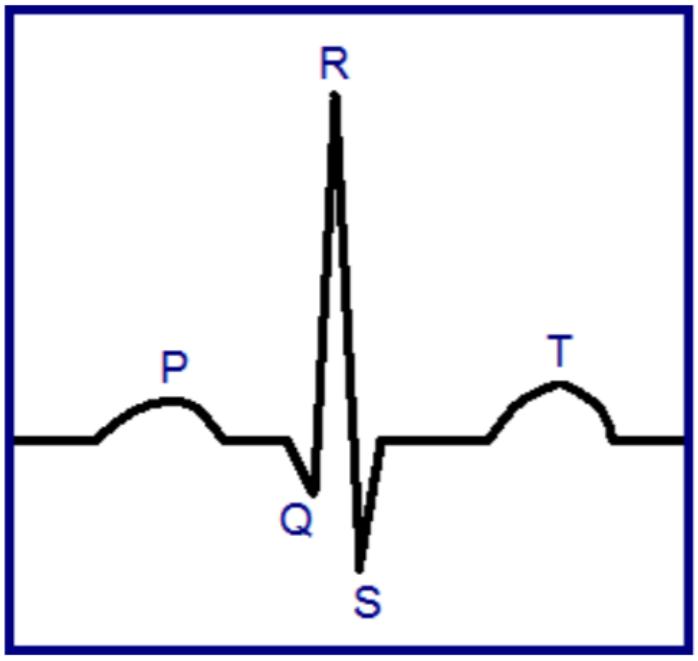


Рисунок 2.2 — Сигнал електрокардіограми

Також є одна хвиля реполяризації передсердь, що має вигляд подібний до зворотної хвилі P, але вона перекривається комплексом QRS. Таким чином, електрокардіограма є надзвичайно корисною для діагностування різноманітних серцевих порушень: шлуночкова гіпертрофія, інфаркт міокарда, порушення електролітів, порушення провідності та інші.

2.4 Системи передачі ЕКГ сигналів портативним мобільним пристроям

На разі кількість систем, що підтримують передачу сигналів електрокардіографії мобільним пристроям є досить невеликою. У випадку усіх відповідних програмних забезпечень підключення відбувається за допомогою технології Bluetooth.

Одним з прикладів таких систем є Cardiospy Mobile ECG (рисунок 2.3). Безумовними плюсами Cardiospy Mobile ECG є те, що користувач має змогу зберігати свої результати ЕКГ у програмі та передивлятися у разі необхідності. Також досить важливою є функція відправки результату за допомогою e-mail, що

робить процедуру обміну результату електрокардіографії користувача з його доктором ще зручнішою. Тим не менш незважаючи на те, що додаток не потребує оплати для скачування, не всі користувачі можуть їм користуватися, оскільки цей додаток потребує виключно прилад цієї компанії (а саме EC-12RM). На жаль, потрібний прилад неможливо купити на даний момент. Таким чином, даний додаток є неможливим для користування через обмеження в приладах, які можуть бути підключені задля передачі ЕКГ сигналів.

Рисунок 2.3 — Відображення мобільного додатку Cardiospy Mobile ECG

Подібним до додатку Cardiospy Mobile ECG є додаток Kardia від компанії AliveCor (рисунок 2.4). Для використання цього додатку також необхідне придбання персонального ЕКГ монітору KardiaMobile. KardiaMobile знімає ЕКГ медичного рівня у будь-якому місці та у будь-який час за 30 секунд, надає інформацію про пульс користувача та може виявити тахікардію, брадикардію та інші порушення серцевого ритму, дозволяє надіслати результати лікарю за

допомогою електронної пошти. Цей додаток є досить популярним – на разі у ньому зареєстровано 60 мільйонів ЕКГ. Отримання результатів електрокардіограми відбувається за допомогою розташування пальців рук на приладі Kardia. Основним недоліком є те, що потрібній для використання додатку прилад є досить дорогим для українського користувача та його придбання можливе виключно на території Сполучених Штатів Америки.

Рисунок 2.4 — Відображення мобільного додатку Kardia та пристрою KardiaMobile

Наступним прикладом є Cardiolyse (рисунок 2.5). Завантаження цього додатку також є безкоштовним і, на відміну від минулих додатків, до Cardiolyse можливо підключити будь-який прилад з потрібними для аналізу функціями. Тим більше, відповідно до розробників, цей додаток крім ЕКГ також демонструє варіабельність серцевого ритму, що також допомагає оцінити стан здоров’я серця. Користувач має також можливість оцінити його настрій та вести таким чином щоденник настрою. Тим не менш, додаток ще знаходиться на етапі розробки та працює неідеально – на даний час користувач не має змоги зареєструватися, що робить усі функції недоступними оскільки тільки зареєстрований користувач має змогу вести щоденник, проводити дослідження ЕКГ та варіабельності серця.

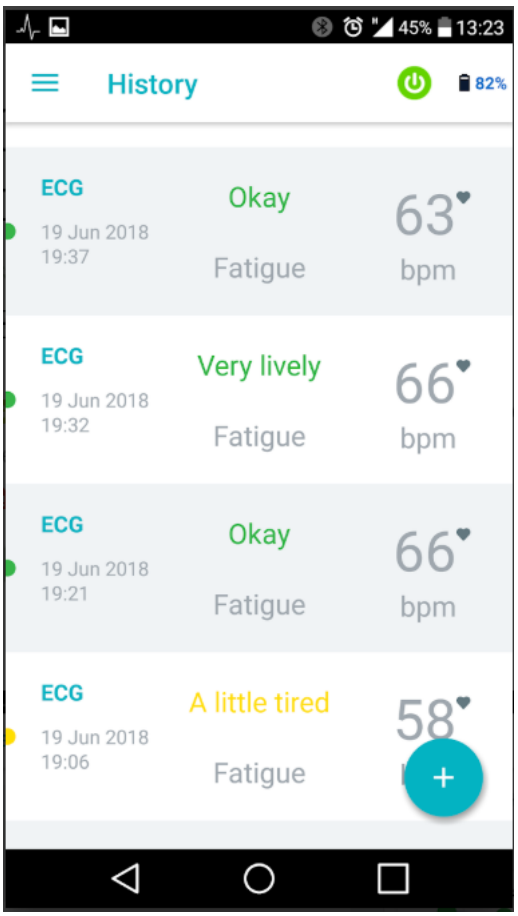


Рисунок 2.5 — Відображення мобільного додатку Cardiolyse

Ці додатки також мають один великий спільний недолік – вони надають усю інформацію виключно на англійській мові. Слід зазначити що на російсько або на українськомовному сегментах взагалі не існує подібних систем навіть на етапі розробки.

Останнім прикладом відповідної системи є додаток Health, що має змогу демонструвати ЕКГ результати користувача виключно за допомогою розумних часів Apple Watch 4 або 5 серії (рисунок 2.6). Ці часи мають велику кількість плюсів – наприклад, вони фіксують кількість витрачених користувачем калорій за допомогою будь-якого типу активності що допомагає притримуватися здорового способу життя, попереджають користувача коли він знаходиться поряд з занадто великим шумом що може бути шкідливим для його здоров’я, нагадують повільно дихати кожну годину для заспокоєння. Як і усі розглянуті раніше додатки, користувач може побачити у Health за допомогою Apple Watch також його пульс. Тим не менш, розумні часи мають досить велику ціну і, незважаючи на доступ

відповідної функції у США та багатої кількості інших країн, функція отримання ЕКГ досить недоступна в Україні. Таким чином, легальним способом неможливо використовувати відповідну функцію в Україні на даний момент.

Рисунок 2.6 — Відображення мобільного додатку Health та розумних часів Apple Watch 5

Таким чином, процедура слідкування пацієнтом за ЕКГ за допомогою потрібного для виміру приладу та додатку є проблемою, особливо в Україні. Ті системи, що мають усі потрібні функції досить знаходяться у процесі розробки, а ті що вже готові для використання, потребують особливий прилад або досить не доступні в Україні, що підтверджує необхідність розробки відповідного програмного забезпечення.

2.5 Висновки до розділу

У даному розділі було розглянута проблема розповсюдженості серцево-судинних захворювань, досліджені типи ССЗ та найпопулярніші з них разом з типами діагностики що використовується на даний час задля діагностування

серцево-судинних захворювань. Тим більше, було досліджена історія виникнення електрокардіографії та розвиток цього методу, а саме були розглянуті декілька інноваційних відкриттів у цій сфері, одним з яких, безумовно, є створення портативних невеличких приладів що дозволяють користувачам вимірювати електрокардіограму без відвідування будь-якого медичного закладу та проводити нагляд за своїм здоров’ям вдома.

Був проведений аналіз вже існуючих відповідних систем для передачі ЕКГ сигналів портативним мобільним приладам, що довів відсутність такого програмного забезпечення та необхідність його розробки оскільки ЕКГ є одним з найважливіших типів діагностики серцево-судинних захворювань.

3 ЗАСОБИ РОЗРОБКИ

Важливим етапом створення будь-якого програмного продукту являється вибір засобів розробки, що надали б змогу реалізувати увесь потрібний функціонал та полегшити етап програмування завдяки зручному інтерфейсу та вже існуючим технологіям. Таким чином, вірний вибір засобів розробки не тільки полегшує процедуру взаємодії користувача з системою, а й робить розробку програмного забезпечення більш ефективною та швидкою.

Як середовище розробки було обрано Android Studio 3.6.3, що є засобом розробки мобільних додатків для смартфонів що працюють на платформі Android.

Передача даних здійснюється за допомогою технології Bluetooth (а саме BLE – Bluetooth Low Energy). Як невід'ємні складові передачі інформації були розглянуті профіль GAP що здійснює пошук пристроїв для передачі разом з профілем GATT що дозволяє відправляти та отримувати невеликі фрагменти даних.

Алгоритм системи був створений за допомогою мови програмування JavaScript та фреймворку React Native, що дозволяє розроблювати кросс-платформні мобільні додатки. Створення та відображення електрокардіограми у додатку здійснюється за допомогою обширної бібліотеки d3.js.

Що стосується інтерфейсу користувача, він був розроблений за допомогою препроцесора JSX, за допомогою якого відбувається транслювання компонентів до мови гіпертекстової розмітки HTML та таблиці каскадних таблиць CSS.

3.1 Середовище розробки Android Studio

Оскільки наразі операційна система Android є найбільш популярною серед користувачів мобільних телефонів (в Україні 66% користувачів користуються саме телефонами на базі Android, а поширеність інших операційних систем не перевищує 13% - така кількість людей використовує смартфони від фірми Apple [15]), програмний продукт було створено перш за все для користувачів платформи

Android.

Саме середовище Android Studio є номером один для розробників додатків для Android оскільки є офіційним інтегрованим середовищем розробки для відповідної платформи. Android Studio можливо завантажити на Windows, Mac та Linux. Використання відповідного середовища можливе як невеликою командою розробників, так і великими міжнародними організаціями. Цей програмний продукт можна завантажити та використовувати безкоштовно. У Studio містяться інструменти розробки рішень для смартфонів, планшетів, а також нові технологічні рішення для Android TV, Android Auto, Android Wear, Glass тощо. Android Studio засновано інтегрованому Java-середовищі розробки програмного забезпечення – IntelliJ IDEA та містить у собі інструменти розробки та редагування коду. Підтримка розробки додатків на системі Android здійснюється за допомогою системи побудови на основі Gradle (інструмент автоматичної збірки, що був розроблений за допомогою Apache Ant та Apache Maven), інтеграції Github, шаблонів коду та емулятору. Це середовище розробки має досить велику кількість переваг: інструменти для пошуку та ліквідації різноманітних проблем, підтримка хмарової платформи Google Cloud Platform, зручний конструктор інтерфейсу що дозволяє що дозволяє відобразити програмне забезпечення на будь-якому приладі, велику кількість літератури та вбудований комплекс засобів SDK (software development kit) [16]. Android Studio сумісна з платформою Google App Engine задля забезпечення швидкої інтеграції в хмарі нових API та функцій. Таким чином, у середовище можна знайти велику кількість різноманітних API – Android Pay, Google Play, Health та інші. Тим більше, Studio забезпечує підтримку всіх платформ Android починаючи з версії 1.6.

Типовий проект в Android Studio має два кореневих каталоги: app та Gradle Scripts. Папка app, у свою чергу, містить папку manifests що має файли конфігурацій або файли маніфесту додатка. Наступний каталог має назву java та містить вихідний код програми. Останньою папкою є каталог res, що складається з файлів які використовуються у додатку (стилі, картинки тощо). Файл AndroidManifest.xml, що міститься у відповідному каталозі, є одним з

найважливіших файлів у проекті та складається з інформації стосовно пакетів додатку, компоненту типу Activity, Service тощо. Саме завдяки цьому файлу надається дозвіл системі на використання або доступ інших компонентів. Каталог res, що містить усі зображення, анімації, звукові файли та інші ресурси що використовуються у проекті, має досить велику кількість підкаталогів, а саме: папку drawable що містить виключно зображення, layout, values та menu що мають xml файли (каталог layout містить xml файли що використовуються з ціллю розробки інтерфейсу користувача, menu складається з xml файлів які використовуються у меню, а xml файли у папці values представляють собою прості значення такі як строки, масиви, стилі тощо) , mipmap що складається зі значків додатку.

3.2 Технологія Bluetooth Low Energy

Bluetooth – це технологія бездротового зв’язку, передача даних у якій здійснюється за допомогою радіоканалів. Основною метою Bluetooth є забезпечення зв’язку між різними типами електричних приладів, до того ж велика увага приділяється саме невеликому розміру електронних компонентів, завдяки чому ця технологія доступна навіть у таких приладах як наручні годинники. Прилади, що працюють на стандарті Bluetooth, здатні здійснювати передачу даних зі швидкістю до 720 кбіт/с на відстані до десяти метрів та передачу трьох голосових каналів. У разі відстані менше 10 метрів пристрій автоматично змінює потужність передачі до потрібного рівня.

Технологія працює за принципом FHSS (Frequency-Hopping Spread Spectrum), що означає розбиття передавачем даних на пакети та їх відправки за допомогою псевдовипадкового алгоритму стрибкоподібної перебудови частоти (1600 разів у секунду) або шаблону що складається з 79 підчастот [17]. Таким чином, передача даних доступна виключно для приладів, що налаштовані на одну частоту, а в іншому разі інформація буде звичайним шумом.

Що стосується саме Bluetooth Low Energy, суть цього протоколу полягає у

передачі даних маленькими пакетами за необхідністю та вимкнення передачі після цього, що і забезпечує невеликі енергозатрати. На відміну від класичного Bluetooth, прилади що використовують саме BLE встановлюють зв’язок виключно у разі необхідності передачі або отримання даних. Bluetooth Low Energy споживає у 10-20 разів менше енергії та здатний передавати дані у 50 разів швидше при порівнянні з класичним Bluetooth [18]. BLE має достатню кількість безумовних переваг: стійкість та надійність, сумісність пристроїв, відкритість та доступність. Технологія адаптивної стрибкоподібної перебудови частоти, що використовується у BLE, дає змогу пристроям що працюють на відповідному протоколу швидко змінювати робочу частоту в широкому діапазоні робочих частот. Таким чином забезпечується зменшення або повне уникнення переповнення у робочому частотному діапазоні. Що стосується надійності, саме захищеність даних завжди була одним з найважливіших пріоритетів при розробці бездротових технологій передачі даних. Bluetooth Low Energy об’єднує одразу декілька механізмів задля забезпечення захищеності даних, а саме: авторизація, аутентифікація, шифрування та алгоритми для боротьби з захопленням даних. Оскільки технологія BLE розроблена спільнотою Bluetooth SIG за принципом відкритого стандарту з суворим дотриманням вимог по кваліфікаційним випробуванням та тестуванням на електромагнітну сумісність, розробники пристроїв що працюють на відповідній технології мають можливість скористатися усіма перевагами BLE, а користувачі отримують максимум комфорту від використання цих пристроїв незалежно від виробника. Саме таким чином забезпечується сумісність пристроїв. Використання технології BLE у загальносвітовому масштабі є можливим завдяки роботі Bluetooth Low Energy на радіохвильовому діапазоні 2.4 ГГц. Пристрої, що працюють на BLE, поділяються на пристрої Single-mode та Dual-mode. Single-mode пристрої підтримують виключно технологію Bluetooth Low Energy та оптимізовані для додатків зі зниженим енергоспоживанням, невеликими габаритами та низькою вартістю. Dual-mode пристрої, у свою чергу, підтримують і BLE, і класичну технологію Bluetooth для забезпечення зворотної сумісності з усіма попередніми версіями специфікації Bluetooth.

Знаходження пристроїв здійснюється за допомогою профілю спільного доступу – GAP (Generic Access Profile), а обмін даними між приладами здійснюється за допомогою профілю спільних атрибутів – GATT (Generic Attribute Profile). Існує два механізми, які можуть використовуватися пристроєм з технологією BLE задля спілкування із зовнішнім світом: трансляція та з'єднання. Саме ці механізми містить у собі GAP. Таким чином, GAP визначає як пристрої з підтримкою BLE можуть зробити себе доступними та як два пристрої будуть спілкуватися один з одним. У випадку трансляції існує транслятор що віщає пакети даних публічно та спостерігач що прослуховує дані в рекламних пакетах які надсилаються мовником. Таким чином, ніякого зв'язку між мовником та спостерігачем не відбувається. У разі з'єднання пристрої явно з'єднуються задля передачі даних. При з'єднанні пристрої поділяються на периферійні та центральні. Спочатку периферійний пристрій рекламує свою присутність задля встановлення з'єднання центральними пристроями. Після підключення периферійні пристрої припиняють передавання даних на інші центральні пристрої та залишаються на зв'язку з пристроєм що прийняв запит з'єднання. Що стосується центральних пристроїв, вони ініціюють з'єднання з периферійними пристроями за допомогою прослуховування рекламних пакетів. Центральний пристрій може підключатися до великої кількості периферійних пристроїв та надсилає запит пакета даних про з'єднання коли хоче підключитися. Таким чином, якщо периферійний пристрій приймає запит від центрального, встановлюється з'єднання. GATT визначає структуру даних та режим роботи, за якими здійснюється взаємодія двох пристроїв що працюють на BLE та обов'язковим для відправки або отримання невеликих частин даних, що мають у BLE назву «атрибути». Як і у разі з GAP, GATT також має деякі ролі, а саме: клієнт та сервер. Клієнт зазвичай надсилає запит на сервер GATT та може читати та записувати атрибути, знайдені на сервері. Сервер зберігає атрибути та робить їх доступними після відправлення запиту клієнтом.

3.3 Мова програмування JavaScript

JavaScript – це кросс-платформна, об’єктно-орієнтована та однопоточна мова програмування високого рівня, що була розроблена компанією Netscape Communication Corporation. JavaScript була створена у 1995 році та на разі є однією з найпопулярніших мов програмування. Спочатку ця мова програмування не була такою потужною оскільки в основному використовувалася для різноманітних анімацій, відомих на той час як DHTML. Програми на цій мові програмування називаються скриптами та виконуються як простий текст – без спеціальної підготовки або компіляції для запуску. Скриптові мови – це мови програмування, що використовуються для автоматизації процесів, які в іншому випадку користувачі повинні виконувати самостійно. JavaScript надзвичайно популярна через повну інтеграцію HTML/CSS та підтримки усіма основними браузерами [19]. Тим більше, програмне забезпечення що написане на JS дійсно є зручним для користувачів через негайний відгук до відвідувачів – їм не потрібно чекати перевантаження сторінки для того щоб побачити чи не забули вони ввести потрібну інформацію. JavaScript також дозволяє підвищити інтерактивність оскільки розробники мають можливість створювати інтерфейси, що реагують навіть при наведенні користувачем миші та зробити інтерфейси більш багатшими за допомогою компонентів перетягування або повзунків. Незважаючи на те, що спочатку ця мова програмування створювалася виключно задля роботи з браузером, на разі вона використовується на великій кількості інших платформ. Для того, щоб код інтерпретувався браузером необхідно включити скрипт у HTML файл або зробити на нього посилання.

Принцип роботи JavaScript у веб-браузері продемонстровано на рисунку 3.1.

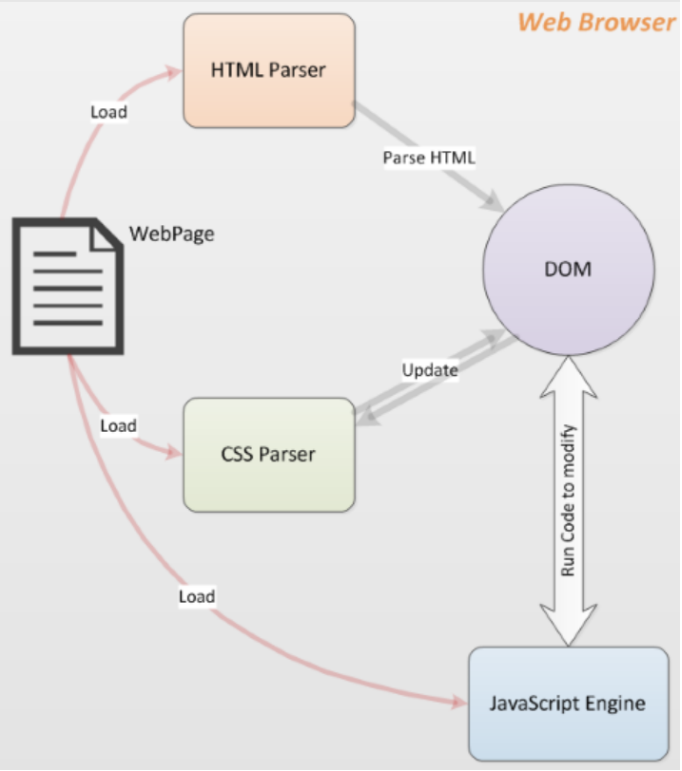


Рисунок 3.1 — Принцип роботи JavaScript у веб-браузері

Спочатку веб-браузер завантажує веб-сторінку, аналізує HTML та створює DOM (об’єктна модель документу) з контенту. Після цього браузер захоплює все, що пов’язане з HTML (наприклад, файли CSS або зображення). Інформація про CSS надходить з аналізатора CSS. HTML і CSS збираються DOM для того щоб створити веб-сторінку першою. Потім завантажуються файли JavaScript та вбудований код, але код не запускається відразу, це відбувається після повного завантаження HTML та CSS. Код JavaScript виконується в порядку запису коду. Це призводить до того, що DOM оновлюється кодом JavaScript та надається браузером. Порядок є надзвичайно важливим оскільки в іншому разі неможлива зміна DOM елементів.

Таким чином, JavaScript використовується для:

– додання інтерактивності на веб-сайти;
– розробки мобільних додатків;
– створення ігор на основі веб-браузера;
– розробка серверної частини веб-сайтів.

Візуалізація електрокардіограми відбувається з допомогою великої

бібліотеки D3.js (Data-Driven Documents), за допомогою якої відбувається обробка та візуалізація даних у мові JavaScript [20]. На відміну від інших подібних бібліотек, D3 базується на SVG та CSS у той час як інші бібліотеки використовують елемент canvas та його можливості замість стандарту SVG. Таким чином, ця бібліотека використовує вектори замість пікселів, що дозволяє створення структур з насиченою графікою та анімацією. Тим більше, використання саме SVG дозволяє зробити файли зображень більш легкими за розміром. D3 дозволяє прив’язувати довільні дані до об’єктної моделі документа (DOM), а потім застосовувати керовані даними перетворення до документа. Таким чином, бібліотека D3 робить можливим, наприклад, створення HTML таблиці з масиву чисел або навіть використовування тих самих даних задля створення інтерактивної смугової діаграми SVG з плавними переходами та взаємодією. Можливість представлення будь-якої відомої структури у вигляді дерева вузлів, кожен з яких представляє набір даних та які пов’язані між собою відношенням «батько-нащадок», дозволяє програмам та скриптам отримувати доступ до веб-документів, змінювати їх зміст, оформлення та структуру. З мінімальними витратами D3 є надзвичайно гнучкою бібліотекою що підтримує великі набори даних разом з динамічною поведінкою задля забезпечення взаємодії та анімації.

D3 має чотири основні принципи роботи:

– поступовий перехід будь-якого атрибуту до іншого (наприклад, зміна кольору тексту);

– можливість вибору необхідного набору DOM-вузлів задля подальшого використовування операторів перетворення цих даних;

– прив’язування даних до відповідних SVG-об'єктів з відповідними параметрами (колір, форма, значення) та поведінкою (переходи, події);

– додавання на сторінку DOM-вузлів, що забезпечує надання сторінці або додатку бажаного зовнішнього вигляду.

React Native – це фреймворк JavaScript, що використовується задля написання мобільних додатків для Android та iOS, завдяки чому розробник має можливість написати код тільки один раз та запустити додаток на будь-якій з цих

двох платформ. Це суттєво спрощує створення програмного продукту оскільки у цьому разі не потрібно тратити гроші та час на два проекту розробки. Цей фреймворк заснований на JavaScript бібліотеці від компанії Facebook – React. Подібно до React, програми на React Native записуються за допомогою суміші JavaScript та XML розмітки (JSX). Саме за допомогою JSX розробник має можливість писати HTML елементи у JavaScript, тобто JSX конвертує HTML теги у React елементи. Цей препроцесор базується на ES6. Що стосується ES6, ця специфікація була створена організацією ECMA International, яка займається стандартизацію інформаційних та комунікаційних технологій. Спочатку мова програмування JavaScript створювалася як скриптова мова для Netscape (американська корпорація в IT індустрії), після чого він був відправлений саме в ESMA International для стандартизації, що призвело до створення стандарту з назвою ECMAScript (ES). Наступні версії JavaScript базувалися вже на ECMAScript. Таким чином, JavaScript є найпопулярнішою реалізацією стандарту ECMAScript. Для візуалізації React Native викликає відповідні API у Objective-C (для iOS) або у Java (для Android). Той факт, що React Native працює використовуючи стандартні API візуалізації своєї хост-платформи є великою перевагою при порівнянні з іншими методами кросс-платформної розробки (наприклад, Cordova) оскільки ці методи розробки мобільних додатків мають меншу ефективність. Також цей фреймворк викликає JavaScript інтерфейси для API платформ, що робить можливим доступ до камери телефону чи, наприклад, місцезнаходження користувача.

Тестування було зроблено за допомогою Jest та Enzyme. Jest – це відкрита JavaScript бібліотека для тестування, що була розроблена компанією Facebook. Jest має велику кількість переваг при порівнянні з іншими засобами для тестування: швидкість та надійність оскільки виконування тестів здійснюється у паралельних потоках та зміна черги запуску у залежності від того як довго виконується кожний тест, надавання інформації про покриття коду тестами за запитом, легкий мокінг – імітування потрібних для тестування об’єктів та детальний опис помилки у разі неуспішного тестування. Enzyme – це спеціальна утіліта, що була розроблена

компанією Airbnb та досить часто використовується для тестування додатків що написані на React оскільки містить корисні для тестування компоненти. Без Enzyme процес тестування був би набагато складнішим оскільки у цьому разі розробнику доведеться самостійно реалізувати багату кількість необхідних функцій. Enzyme суттєво спрощує маніпулювання та переміщення React компонентів. Основними функціями є shallow та mount. Функція shallow завантажує у пам’ять лише кореневий компонент, а mount завантажує повне дерево DOM.

3.4 Мова гіпертекстової розмітки HTML

HTML (HyperText Markup Language) – це мова розмітки гіпертексту, що дозволяє створити та структурувати розділи, заголовки, посилання тощо для додатків та веб-сторінок. Ця мова розмітки є невід’ємною складовою та основою переважної більшості веб-сторінок. Таким чином, HTML дозволяє організовувати документи подібно до Microsoft Word, але не є мовою програмування, тобто не має можливості створювати динамічні функції. Для роботи з HTML використовуються теги та атрибути, які дозволяють розмітити сторінку сайту. Теги – це спеціальні маркери, що інтерпретуються браузером. Атрибути – це складові тегів, що використовуються для розширення можливостей окремих тегів та більш гнучкого управління вмістом контейнерів. Саме за допомогою атрибуту можливо вирівняти абзац або зображення усередині тега. Також використовуються значення що присвоюються атрибутам та задають саме параметри вирівнювання або інших змін. Значення можуть бути текстовими (наприклад, right) або числовими (у цьому разі задається, наприклад, висота зображення у пікселях). У разі якщо у тегу не встановлений будь-який з атрибутів, браузер використовує значення, що встановлене за замовченням.

Таким чином, кожна HTML сторінка містить у собі набір тегів, що створюють ієрархію завдяки якій користувач бачить контент структуровано – за розділами, параграфами та іншими блоками. Слід зазначити, що більшість

елементів HTML мають відкриваючий (<tag>) та закриваючий (</tag>) тег. Усі HTML документи можна умовно поділити на дві частини – оголошення типу документа (Document Type Declaration) та сама HTML-сторінка. Кожна зі сторінок мови розмітки гіпертексту у свою чергу обов'язково містить 3 теги - <html> (елемент найвищого рівня, що охоплює кожну HTML сторінку та знаходиться після оголошення типу документа), <head> (містить заголовок сторінки, кодування та іншу службову інформацію яку не бачить користувач), <body> (містить всю інформацію, що відображається на сторінці). Різноманітні розділи оформлюються за допомогою тегу .

Теги в HTML поділяються на дві групи: блокові та вбудовані. Блокові елементи завжди починаються з нового рядка та займають увесь наявний простір на ньому. Прикладами блокових елементів можуть послугувати заголовки та абзаци. Вбудовані (строкові) елементи у свою чергу не починають новий рядок на сторінці та займають виключно стільки місця на рядку скільки потрібно. Зазвичай вони використовуються задля форматування внутрішнього елементу на рівні блоків. Прикладами вбудованих тегів є посилання та підкреслення рядку. Веб-сторінки, що розроблені за допомогою HTML мають вагомі переваги при порівнянні з іншими методами розробки (наприклад, Wordpress) оскільки вони набагато швидше працюють та економлять трафік на хостингу. Минула версія HTML з'явилася у 1999 році та має увесь потрібний базовий функціонал задля розробки веб-сторінок, а обновлення HTML 5, що було розроблене у 2014 році, зробило можливим використання векторної графіки, баз даних SQL та багато іншого що суттєво покращило процес розробки.

HTML має відкритий код та є безкоштовним, підтримується Всесвітнім консорціумом веб-сторінок (W3C) та запускається у кожному веб-переглядачі. Використання виключно HTML застосовується для статичних веб-сторінок оскільки не дозволяє розробнику реалізувати логіку та створювати динамічну функціональність, що призводить до обов'язкового використання JavaScript або інших мов програмування у цьому випадку. Усі веб-сторінки необхідно створювати окремо навіть у разі використання однакових елементів.

3.5 Таблиця каскадних стилів CSS

CSS (Cascading Style Sheets) – це мова таблиць стилів, завдяки якій здійснюється надання стилів структурованим документам (наприклад, HTML-сторінками). CSS є стандартизованою у веб-браузерах відповідно до специфікації W3C. Стиль – це сукупність правил, що використовуються по відношенню до елементу гіпертексту та визначають його спосіб відображення. Прикладами стилів можуть слугувати шрифт або колір тексту. CSS суттєво спрощує процес створення веб-сторінок оскільки відділяє вміст документів від їх стилю подання. Оголошення стилю завжди складається з селектору та опису, що завжди укладений у фігурні скобки .

Селектор – це елемент, до якого будуть застосовуватися стилі. Він може бути універсальним, за типом елемента, за класом, за ідентифікатором та за атрибутом [21]. Тим більше, селектори можуть групуватися за допомогою коми, пробілу тощо. Що стосується опису, він завжди складається з властивості та значення. Властивість визначає одну або декілька характеристик селектора та визначає формат його відображення (розмір, шрифт , відступ). Значення, у свою чергу, являє собою фактичні числові чи строкові константи.

Каскадна таблиця стилів може використовуватися на веб-сторінках трьома різними способами: за допомогою вбудовування, що дозволяє використовувати стиль до заданого тегу, за допомогою впровадження яке робить можливим управління стилями сторінки ціликом (написання CSS інструкцій у заголовку певної .html сторінки) та за допомогою зв’язування (у цьому разі опис усіх стилів виноситься до окремого файлу, посилання на яке залишається на потрібній сторінці сайту, що дозволить зв’язати файл .html із зовнішньою таблицею стилів). CSS має достатню кількість переваг: підвищення гнучкості та зручності редагування завдяки тому що для змінення оформлення документу достатньо лише зробити правку у CSS, спрощення коду через зменшення повтору елементів, можливість легко застосовувати до одного документу різних стилів (наприклад,

створення адаптивної версії для мобільних пристроїв), спрощення процесу розробки через можливість застосування одних і тих же стилів до різних сторінок додатку та прискорення процесу загрузки сторінок оскільки CSS має можливість кешуватися при першому відкритті, а після завантажуються тільки дані та структура.

Таким чином, CSS використовується для керування шрифтами, кольорами, надання курсиву або підкреслювання, контролювання різноманітних частин сторінки (наприклад, заголовок, колонтитул, тіло, розділи тощо), що є надзвичайно корисним у разі коли вміст потрібно по-різному відобразити залежно від пристрою користувача (комп’ютер, планшет, смартфон) та надання більш приємного для користувача інтерфейсу.

3.6 Висновки до розділу

В даному розділі були розглянуті використані засоби розробки. Таким чином, було проаналізоване середовище розробки Android Studio, типова структура проекту у відповідному середовищі, технології Bluetooth та Bluetooth Low Energy разом з профілем спільного доступу (GAP) та профілем спільних атрибутів (GATT), які забезпечують передачу даних разом, мова програмування JavaScript та її фреймворк React Native, що забезпечили підтримку продукту системою Android. Використання бібліотеки d3.js забезпечило відображення електрокардіограми.

Також були розглянуті мова розмітки гіпертексту HTML та таблиця каскадних стилів CSS, які забезпечили структурне та візуальне оформлення системи за допомогою використання JSX, що робить можливим роботу з відповідними технологіями у мові програмування JavaScript.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Для моніторингу стану здоров’я та своєчасного виявлення можливих проблем з серцево-судинною системою додаток повинен бути одночасно інформативним та інтуїтивно зрозумілим в роботі та способах взаємодії.

Забезпечення підтримки системою платформи Android базується на модернізованому Polar SDK, а розміщення та оформлення елементів виконується з використанням структурних елементів фреймворку React Native. Відображення ЕКГ здійснюється за допомогою бібліотеки d3.js.

Відправка результатів електрокардіограми відбувається за у месенджері або на електронну пошту.

4.1 Структура програмного забезпечення

Для реалізації системи передачі ЄКГ сигналів мобільним пристроям був розроблений мобільний додаток для пристроїв на базі операційної системи Android, а в якості інтерфейсу користувача виступає власноруч змодельований, розроблений та перенесений в додаток web інтерфейс.

Додаток використовує бібліотеку Chart.js для візуального відображення отриманих даних. Бібліотека Chart.js створена на базі потужної та масштабної бібліотеки d3.js, що надає їй широкі можливості для роботи з даними, а також для їх збереження.

Структуру проекту(рис. 4.1) можна розділити на такі блоки: файл index.js, файл App.js, тека node_modules та тека android.

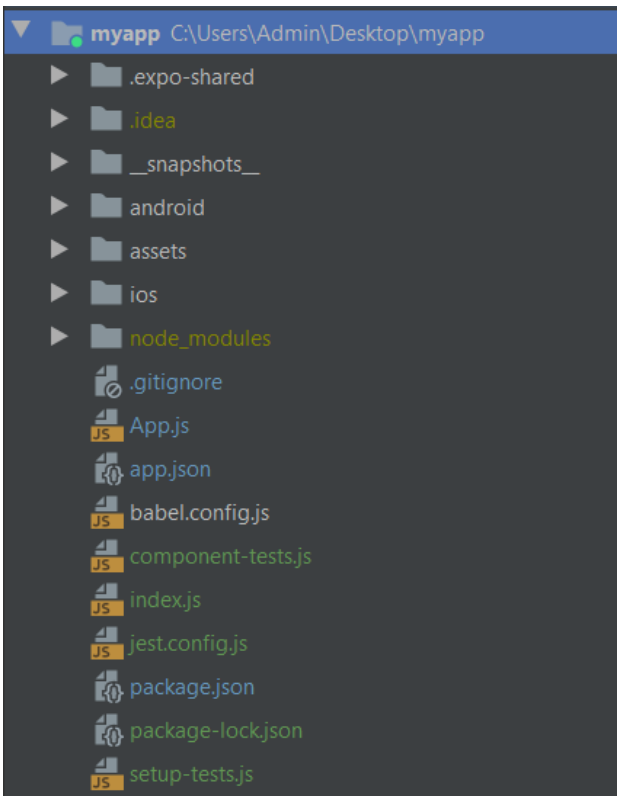


Рисунок 4.1 — Структура файлів проекту

Папка node_modules включає в себе усі необхідні сторонні бібліотеки (у тому числі фреймворк React Native). Використання сторонніх бібліотек при розробці програми дозволяє суттєво прискорити розробку, а також робить можливим використання у проекті оптимізованих та ефективних рішень.

Папка Android складається з файлів SDK, та кінцевих файлів додатку (рисунок 4.2), що отримані у результаті обробки вхідного коду, доповнення його кодом всіх використаних сторонніх бібліотек та приведенням в форму яка буде сприйматися операційною системою Android у якості додатка.

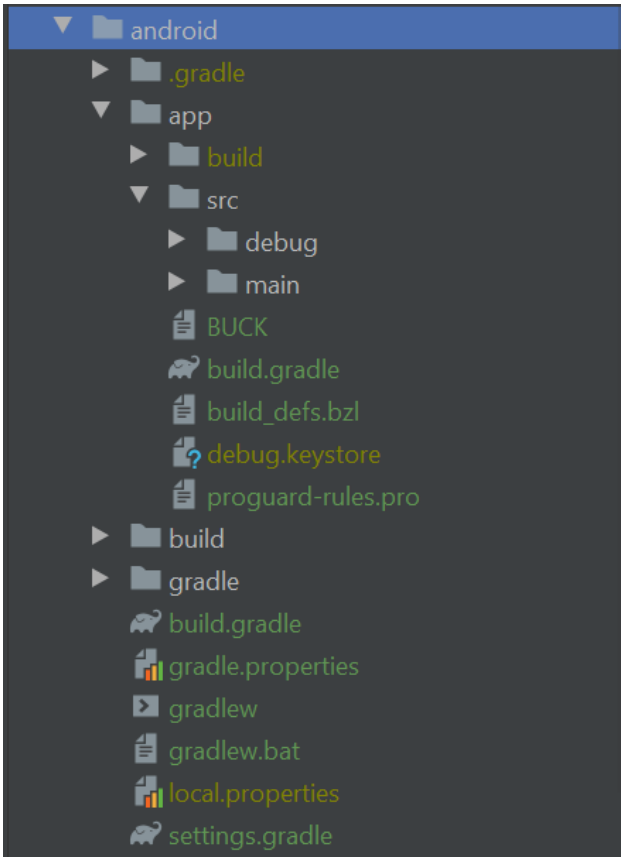


Рисунок 4.2 — Структура теки Android

Файл index.js відповідає за відображення основного програмного компоненту App.js в якості стартової сторінки додатку.

Що стосується файлу App.js, він включає в себе всю логіку та інформаційну структуру додатку, що зображені на рисунку 4.3.

```
const chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';

const Base64 = {...};

function _base64ToArrayBuffer(base64) {...}

const bleManager = new BleManager();
const pmdID = 'fb005c80-02e7-f387-1cad-8acd2d8df0c8';
const pmdControlChar = 'fb005c81-02e7-f387-1cad-8acd2d8df0c8';
const pmdMTUchar = 'fb005c82-02e7-f387-1cad-8acd2d8df0c8';
const bataryId = '180f';

const parseECGData = (data, time = 0)  => {...}

export default function App() {...}

const Button = ({text, clickFunc}) => {...}

const SaveBtn = ({text, imgref}) => {...}

const Popup = ({blockDisplay,blockDisplayFunc,popupText,popupImg}) => {...}

const CloseBtn = ({text,clickFunc}) => {...}

const common = new StyleSheet.create({...})
const styles = new StyleSheet.create({...});
```

Рисунок 4.3 — Структура файлу App.js

Файл можна поділити на такі окремі модулі: модуль декодування, модуль перетворення даних, модуль підключення, модуль візуалізації та файловий модуль.

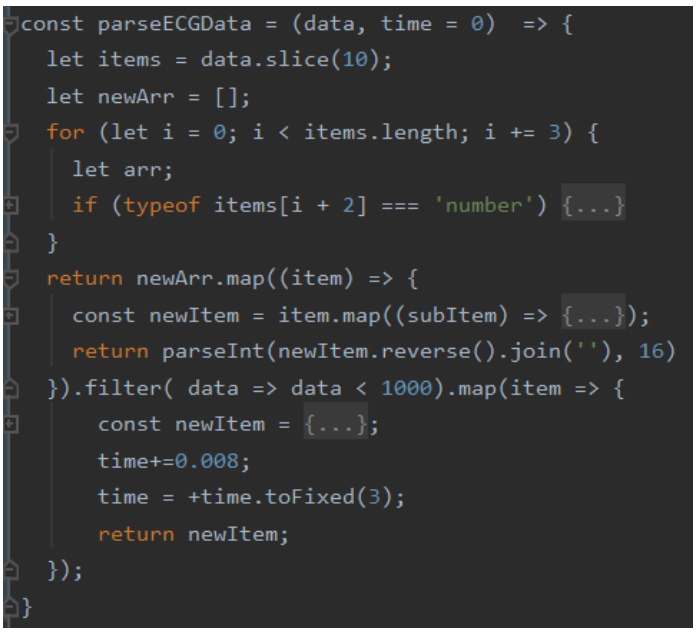
Робота додатку починається з модуля підключення, який представлений всередині компонента App та структура якого зображена на рисунку 4.4. Перед модулем підключення знаходиться потрібний текст задля відображення опису додатку, норм ЕКГ та норм пульсу. Сам модуль починається з процесу сканування доступних Bluetooth пристроїв, після знаходження підходящого пристрою додаток приєднує його до мобільного телефону та проводить запит на отримання всіх доступних на ньому сервісів. Оскільки кожен сервіс може мати декілька характеристик що відповідають за супутню інформацію, після отримання списку сервісів додаток підписується на отримання потоку даних з потрібних сервісів та робить запит на отримання шуканих характеристик. Після завершення процесу отримання значень необхідних характеристик, починають роботу модулі декодування та

перетворення.

```
useEffect(() => {
    if (!device) {
      bleManager.onStateChange((newState) => {
        if (newState === 'PoweredOn') {
          bleManager.startDeviceScan([], {allowDuplicates: true}, async (err, device) => {
            if (device) {
              if (device && device.name && device.name.startsWith('Polar')) {
                bleManager.stopDeviceScan();
                await bleManager.connectToDevice(device.id, {
                  requestMTU: 512
                });
                setDevice(device);
                const scanned = await device.discoverAllServicesAndCharacteristics();
                const data = await scanned.services();
                for (let key in data) {
                  if (data[key].uuid.includes('180f')) {...}
                    if (data[key].uuid.includes('180d')) {...}
                }
                device.monitorCharacteristicForService(pmdID, pmdMTUchar, (err, data) => {
                  if (data && data.value && (data.value).length > 20) {...}
                });
                const res = await device.writeCharacteristicWithResponseForService(pmdID, pmdControlChar, 'AgAAAYIAAQEOAA==');
                 console.log('stream response', _base64ToArrayBuffer(res.value));
              }
            }
          })
        }
      }, true)
    }
}, []);
```

Рисунок 4.4 — Структура компоненту підключення

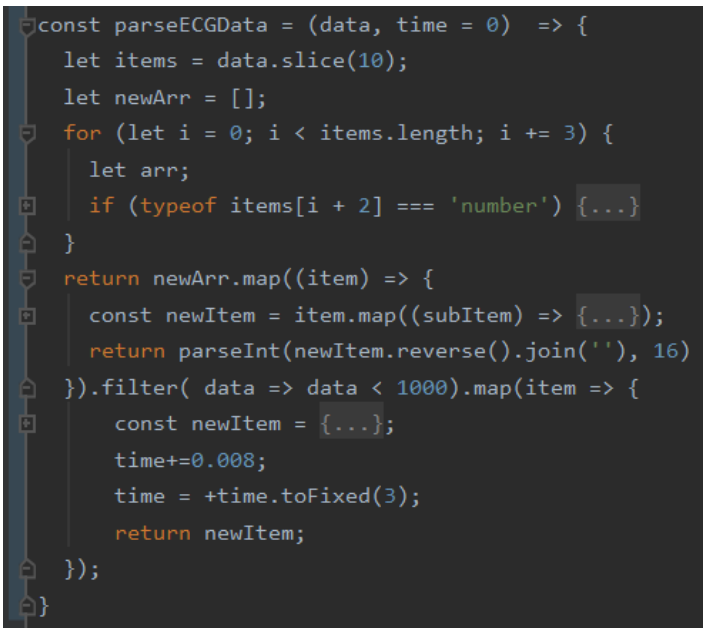
Модуль декодування представлений реалізацією функції розшифрування даних в форматі base64, а також функцією base64ToArrayBuffer. Функція base64ToArrayBuffer містить алгоритм приведення вхідної строки у форматі base64 у формат масиву байтів даних, що представлені числами у десятковій системі числення. Вивід даних заряду батареї та пульсу здійснюється саме у десятковій системі, а відображення електричної активності серця відбувається за допомогою додаткового приведення до шістнадцяткової системи числення задля формування одного значення ЕКГ та подальшого повернення до десяткової системи числення для візуалізації на графіку. Робота з даними ЕКГ відбувається завдяки модулю перетворення даних. Структурний вигляд модулю декодування розміщено на рисунку 4.5.



```
const parseECGData = (data, time = 0)  => {
  let items = data.slice(10);
  let newArr = [];
  for (let i = 0; i < items.length; i += 3) {
    let arr;
    if (typeof items[i + 2] === 'number') {...}
  }
  return newArr.map((item) => {
    const newItem = item.map((subItem) => {...});
    return parseInt(newItem.reverse().join(''), 16)
  }).filter( data => data < 1000).map(item => {
      const newItem = {...};
      time+=0.008;
      time = +time.toFixed(3);
      return newItem;
  });
}
```

Рисунок 4.5 — Структура компоненту декодування

Модуль перетворення даних, структура якого зображена на рисунку 4.6, представлений розробленим алгоритмом обробки вхідного декодованого набору даних ЕКГ. Суть роботи модулю полягає у відокремленні від цілісного масиву даних окремих його частин. Таким чином, набір даних ЕКГ містить: статус пакету даних (2 байти), часовий код останнього заміру в межах цього пакету у наносекундах (8 байт), заміри ЕКГ, що представлені у вигляді послідовно розташованих байтів в пакеті. Кількість замірів та частота надходження пакетів інформації залежить від встановленого максимально допустимого розміру одного пакету. У розробленому додатку встановлене обмеження на отримання 73-х замірів два рази в секунду, кожен з котрих представлений трьома послідовними бітами. Відповідний пакет даних створюється два рази в секунду. Такий розмір пакетів є рекомендованим для оптимального використання обчислювальних ресурсів при роботі з неперервними потоками даних. Після обробки та приведення в десяткову систему числення заміри ЕКГ можуть бути відображені на графіку.



```
const parseECGData = (data, time = 0)  => {
  let items = data.slice(10);
  let newArr = [];
  for (let i = 0; i < items.length; i += 3) {
    let arr;
    if (typeof items[i + 2] === 'number') {...}
  }
  return newArr.map((item) => {
    const newItem = item.map((subItem) => {...});
    return parseInt(newItem.reverse().join(''), 16)
  }).filter( data => data < 1000).map(item => {
      const newItem = {...};
      time+=0.008;
      time = +time.toFixed(3);
      return newItem;
  });
}
```

Рисунок 4.6 — Структура компоненту перетворення даних

Модуль візуалізації, що зображений на рисунку 4.7, представлений кінцевим результатом роботи компоненту App та включає в себе структурне наповнення основної сторінки додатку даними, ініціалізацію графіку для демонстрації замірів ЕКГ, а також динамічне оновлення даних при надходженні нового пакету.

```
return (
    <View style={styles.wrapper} contentContainerStyle={common.containerCenter}>
        <Text style={styles.battery}>
            Заряд: {accLevel}%
        </Text>
        <View contentContainerStyle={common.containerCenter}>
            <Text style={styles.pulse} contentContainerStyle={common.containerCenter}>
                {heartRate}
            </Text>
        </View>
        <View style={styles.btnWrap} contentContainerStyle={common.containerCenter}>
            <Button text={'Опис додатку'} clickFunc={setAppInfoDisplay}/>
            <Button text={'Норми пульсу'} clickFunc={setPulseInfoDisplay}/>
            <Button text={'Норми ЕКГ'} clickFunc={setECGInfoDisplay}/>
            <SaveBtn text={'Поділитися ЕКГ'} imgref={imgRef}/>
        </View>

        // Компонент ScrollView для отримання можливості гортати графік
        <ScrollView horizontal={true} contentContainerStyle={common.containerEnd}>
            // Компонент ViewShot для зазначення області знімку
            <ViewShot collapsable={false} ref={imgRef} snapshotContentContainer={true}>
                <LineChart
                style={{height: 200, minWidth: 100 * dataSet.length/73}}
                contentContainerStyle={{justifyContent: 'flex-end'}}
                data={dataSet}
                keys={['item', 'time']}
                colors={['#8800cc', '#aa00ff']}
                contentInset={{top: 30, bottom: 30}}
                svg={{stroke: '#bd93f9'}}
                xAccessor={(item) => item.item.time}
                yAccessor={(item) => item.item.item}
                numberOfTicks={73}
                >
                <Grid/>
                </LineChart>
            </ViewShot>
        </ScrollView>

        // Компоненти Popup розміщуються в кінці основного компоненту
        // та викликаються при натисканні відповідних кнопок
        <Popup blockDisplay={appInfoDisplay} blockDisplayFunc={setAppInfoDisplay} popupText=
        <Popup blockDisplay={pulseInfoDisplay} blockDisplayFunc={setPulseInfoDisplay} popupT
        <Popup blockDisplay={ECGInfoDisplay} blockDisplayFunc={setECGInfoDisplay} popupText=

    </View>
);
```

Рисунок 4.7 — Модуль візуалізації

Файловий модуль, що представлений окремим компонентом react-native зображеним на рисунку 4.8, реалізовує усі етапи підготовки та відправки файлу електрокардіограми. Спочатку відбувається захват зображення в зоні графіку ЕКГ, у результаті якого отримується зображення у вигляді строки у форматі jpg та його URI. Після цього відбувається перевід формату зображення у base64 задля можливості відправки за допомогою функції share.

```
const SaveBtn = ({text, imgref}) => {
    return (
        <TouchableOpacity style={styles.touchable} activeOpacity={0.9} onPress={() => {
            captureRef(imgref, {
              format: "jpg",
              quality: 0.8
            }).then(
              uri => {
                const image = uri;
                RNFetchBlob.fs.readFile(image, 'base64')
                .then((data) => {
                  let shareOptions = {
                    title: "ЕКГ",
                    message: "Збережене ЕКГ",
                    url: `${data}`,
                    subject: "Збережене ЕКГ"
                  };

                  Share.open(shareOptions)
                    .then((res) => console.log('res:', res))
                    .catch(err => console.log('err', err))
                });
              },
              error => console.error("Щось пішло не так під час процесу відправки ЕКГ", error)
            );
        }}>
```

Рисунок 4.8 — Файловий модуль

Маючи URI, за допомогою RNFetchBlob зображення підготовлюється до відправлення та доповнюється супутніми даними. Після цього, використовуючи функцію share що є імпортованою з модуля react-native-share, відкривається стандартний інтерфейс відправки файлу у системі Android.

Тестування відбувається у файлу component.test.js, структура якого зображена на рисунку 4.9. Задля тестування розробленого програмного продукту використовується тестування за допомогою снепшотів окремих компонентів та тестування функцій. Усього розроблено 8 тестів – 5 з них виконують тестування компонентів (основний компонент, компонент Popup, компонент CloseBtn, компоненти Button та SaveBtn). Що стосується тестування окремих функцій, було протестовано коректне декодування зашифрованої у форматі base64 строки, перевірка на помилку у разі прийманні невірно закодованої строки у форматі base64 та повертання масиву значень функцією _base64ToArrayBuffer.

```
test('Коректний рендер основного компоненту', () => {...});

test('Коректний рендер компоненту Popup', () => {...});

test('Коректний рендер компоненту CloseBtn', () => {...});

test('Коректний рендер компоненту Button', () => {...});

test('Коректний рендер компоненту SaveBtn', () => {...});

function atob(input){...}

function _base64ToArrayBuffer(base64) {...}

const properBase64String = 'c2ltcGxlIHN0cmluZyB0byB0ZXN0IGJhc2U0IHRlc3RpbmcgZ2l0aCBqZXN0';
const notACorrectBase64String = 'c2ltcGxlIHN0cmluZyB0byB0ZXN0IGJhc2U0IHRlc3RpbmcgZ2l0aCBqZXN0k';
const chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';

test('функція atob коректно декодує закодовану у форматі base64 строку', () => {
  expect(atob(properBase64String)).toBe('simple string to test base4 testing with jest');
});

test('функція atob сигналізує про помилку коли приймає некоректно закодовану base64 строку', () => {
  expect(atob(notACorrectBase64String)).toBeFalsy();
});

test('функція _base64ToArrayBuffer повертає масив Uint8 значень', () => {
  expect(_base64ToArrayBuffer(properBase64String)).toBeInstanceOf(Uint8Array);
});
```

Рисунок 4.9 — Структура файлу component-tests.js

Усі тести були успішно пройдені (рисунок 4.10), що підтверджує правильність розробленого програмного продукту.

```
PASS ./component.enzyme.test.js (9.997s)
  √ Коректний рендер основного компоненту (572ms)
  √ Коректний рендер компоненту Popup (5ms)
  √ Коректний рендер компоненту CloseBtn (6ms)
  √ Коректний рендер компоненту Button (3ms)
  √ Коректний рендер компоненту SaveBtn (2ms)
  √ функція atob коректно декодує закодовану у форматі base64 строку (1ms)
  √ функція atob сигналізує про помилку коли приймає некоректно закодовану base64 строку (1ms)
  √ функція _base64ToArrayBuffer повертає масив Uint8 значень

Test Suites: 1 passed, 1 total
Tests:       8 passed, 8 total
Snapshots:   5 passed, 5 total
Time:        11.425s, estimated 13s
Ran all test suites.
```

Рисунок 4.10 — Результат тестування

Таким чином було проведено перевірення правильності усіх компонентів системи, отримання даних у вірному форматі, отримання помилки у разі невірного формату даних, що є надзвичайно важливим у випадку розробки програмного продукту що використовується у медичних цілях.

4.2 Висновки до розділу

У цьому розділі розповідається про реалізацію структури розробленого мобільного додатку для користувачів смартфонів на платформі Android. Завдяки тому, що більшість Bluetooth пристроїв використовують протокол GATT для організації своєї роботи, забезпечується сумісність додатку з будь яким пристроєм які повністю або частково містять потрібні характеристики.

Дані проходять довгий шлях від отримання їх від пристрою в закодованому вигляді до відображення їх на графіку або в якості показника. При роботі над додатком багато уваги приділялося розробці алгоритмів прийому декодування та обробки даних. Тестування було здійснено у вигляді тестування основних компонентів та функцій додатку, перевірку правильності реагування системи на вірно та невірно закодовану інформацію.

У ході роботи додатку більша частина роботи проходить автоматично, без демонстрації користувачу, який отримує лише кінцевий результат.

5 РОБОТА КОРИСТУВАЧА З СИСТЕМОЮ

Розроблений мобільний додаток не потребує підключення до мережі Інтернет. Для доступу необхідно встановити додаток та мати прилад, що має змогу отримувати дані електрокардіограми та пульс користувача.

5.1 Системні вимоги

Для можливості передавати ЕКГ сигнали та імпульс необхідно мати відповідний пристрій з датчиком серцевого ритму, що має змогу підключатися до телефону за допомогою технології Bluetooth. Прикладами відповідного пристрою можуть слугувати нагрудні пульсометри Wahoo Fitness Tickr X, Garmin Hrm Tri, Suunto Smart Belt, Polar H10 та MyZone MZ-3. У ході даної роботи використовувався прилад Polar H10 (рисунок 5.1).

Рисунок 5.1 — Зображення приладу Polar H10

Для можливості інсталювати та працювати з програмним продуктом необхідно мати смартфон з операційною системою Android версії не нижче ніж 8.0 та технологією Bluetooth. Робота з системою здійснюється в офлайн режимі але задля поширення електрокардіограмою необхідний доступ до мережі

інтернет. На разі скачування у Google Play є неможливим, але можливе використання APK (Android Package Kit), що застосовується для встановлення різноманітних додатків на Android-пристроях. Інформація про APK файл подано на рисунку 5.2.

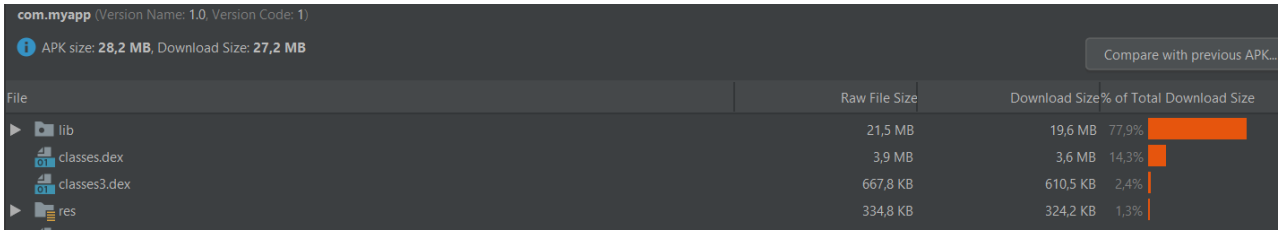


Рисунок 5.2 — Інформація про APK файл

Таким чином, відповідний файл може бути розміщений у мережі інтернет, магазині додатків Google Play та просто відправлений за допомогою будь-якого типу комунікації в інтернеті.

5.2 Робота користувача з програмним продуктом

При відкритті мобільного додатку він має вигляд зображений на рисунку 5.3 до моменту підключення до пристрою що передає дані ЕКГ та пульсу.

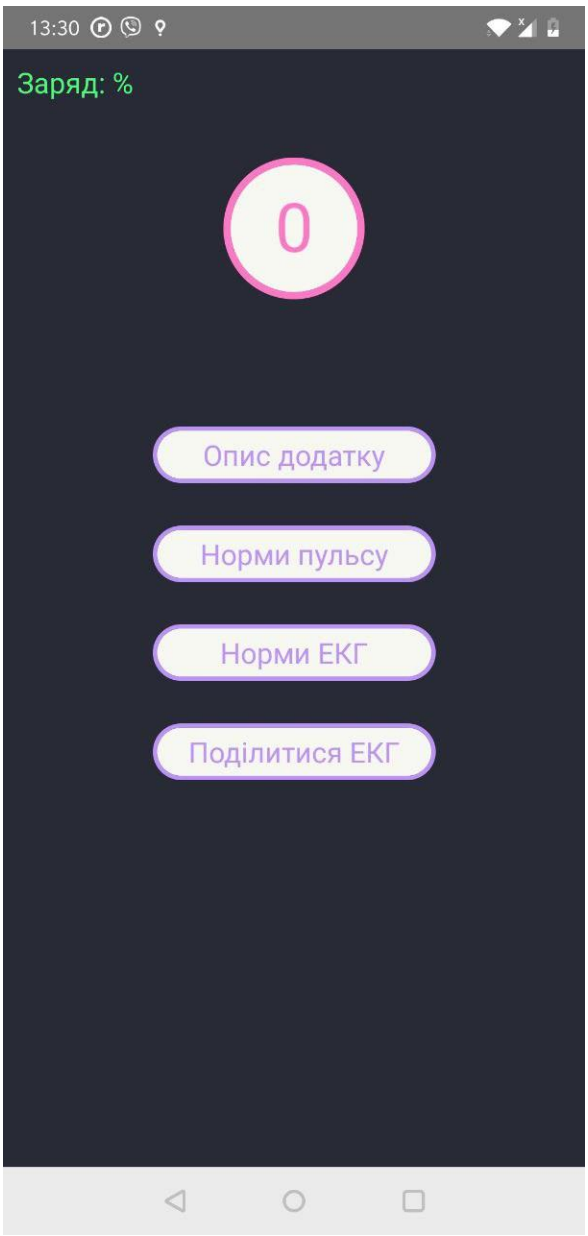


Рисунок 5.3 — Зображення додатку до підключення приладу

Після автоматичного підключення приладу до системи, відображається пульс користувача на даний момент та заряд приладу у відсотках (рисунок 5.4).

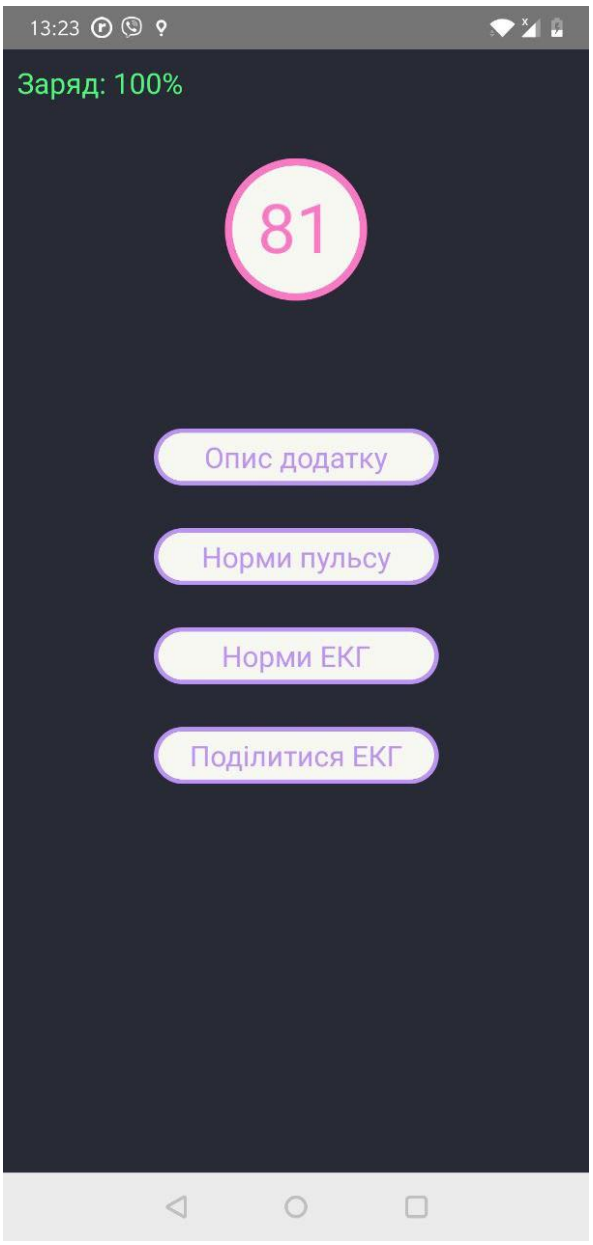


Рисунок 5.4 — Зображення додатку після підключення приладу

Після невеликого відрізку часу користувач вже має змогу побачити свою кардіограму, що зображено на рисунку 5.5.

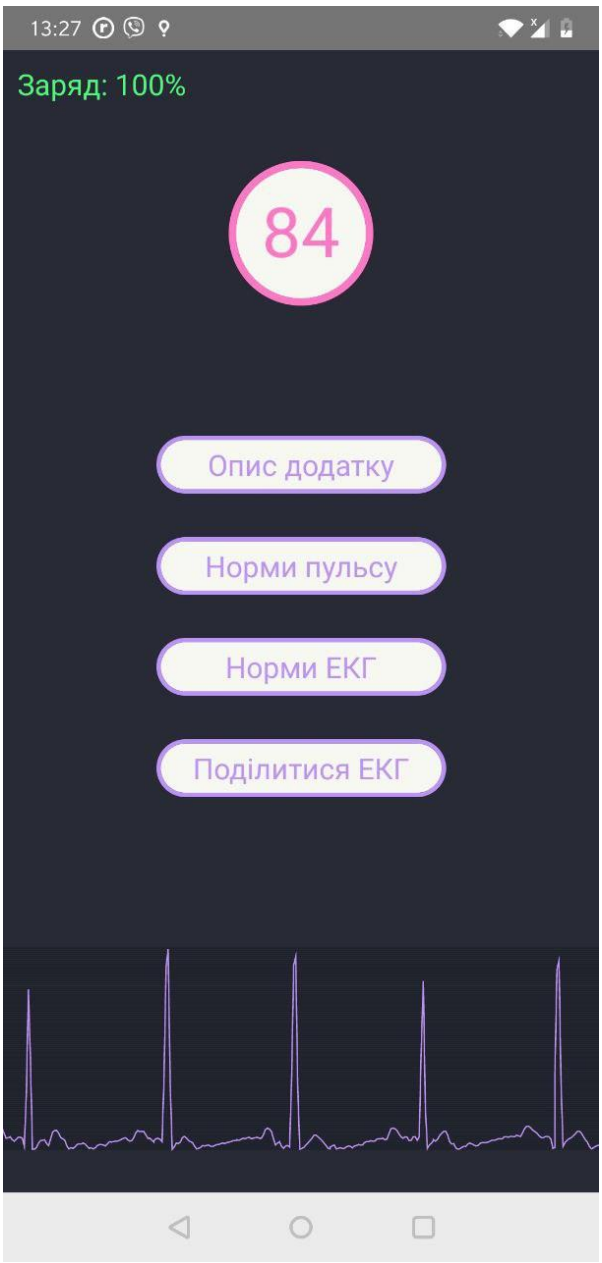


Рисунок 5.5 — Зображення додатку з електрокардіограмою

Користувач має можливість натиснути одну з чотирьох додаткових кнопок. За допомогою кнопки «Опис додатку» відображається сторінка з коротким описом (рисунок 5.6).

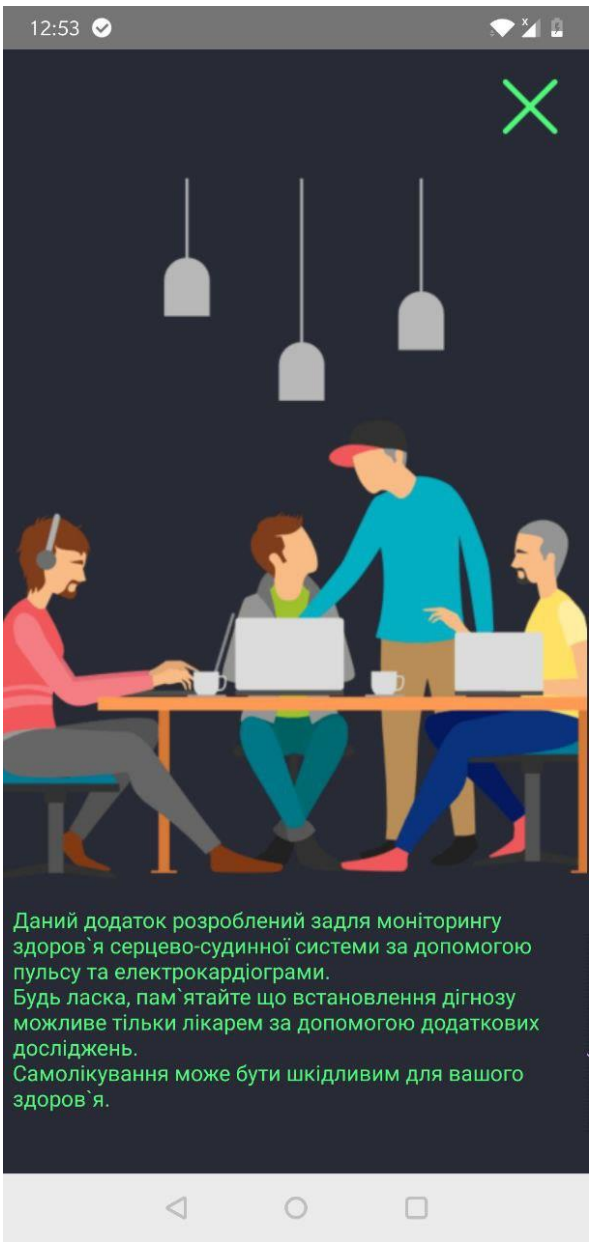


Рисунок 5.6 — Зображення додатку після натискання кнопки «Опис додатку»

Також користувач має змогу продивитися інформацію про норми пульсу та електрокардіограми за допомогою відповідних кнопок у системі.

Таким чином, після вибору кнопки «Норми ЕКГ», відкривається сторінка що зображена на рисунку 5.7.

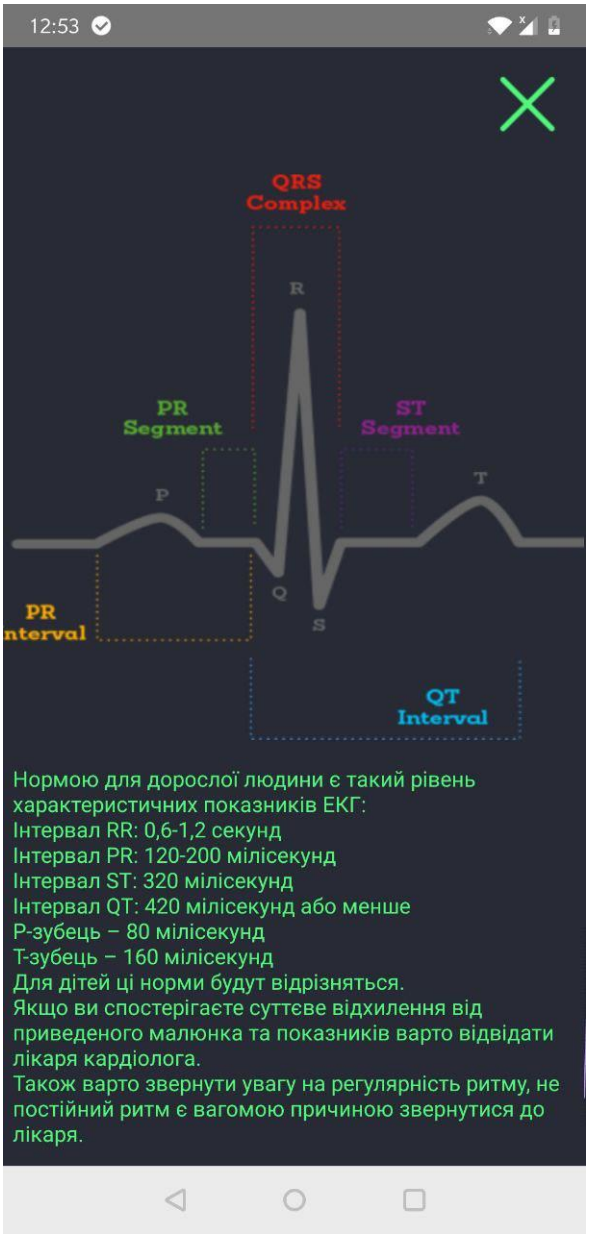


Рисунок 5.7 — Зображення додатку після вибору кнопки «Норми ЕКГ»

За допомогою зеленого хрестику можна повернутися до головного меню та вибрати кнопку «Норми пульсу» (рисунок 5.8).

Рисунок 5.8 — Зображення додатку після вибору кнопки «Норми пульсу»

Користувач має змогу обмінюватися результатами своєї електрокардіограми що забезпечує комунікацію з лікарем. Таким чином, після натискання кнопки «Поділитися ЕКГ» відкривається додаткове вікно, де користувач має змогу обрати потрібний месенджер або спосіб відправки ЕКГ та обрати потрібну людину або вказати його електронну адресу (рисунок 5.9).

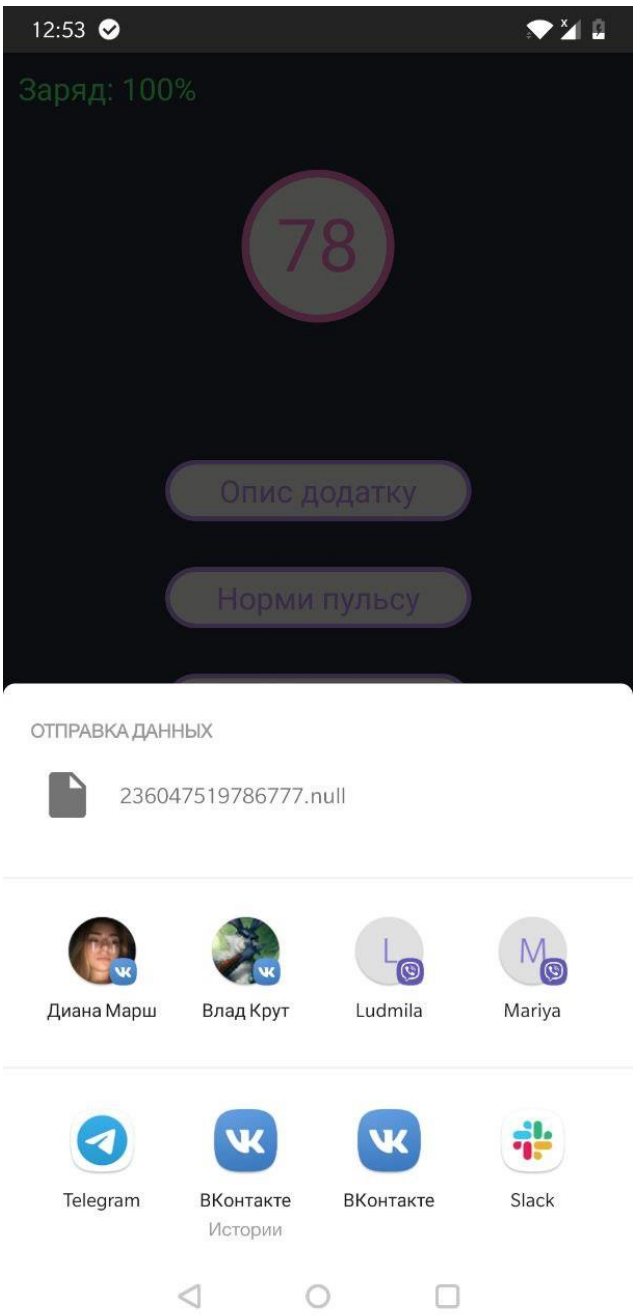


Рисунок 5.9 — Зображення екрану смартфону після натискання кнопки «Поділитися ЕКГ»

Після відправки електрокардіограми отримувач одержує її у вигляді, що зображено на рисунку 5.10. У цьому разі задля відправки результати було обрано месенджер Telegram, але користувач має змогу відправити отриманий результати за допомогою будь-якого іншого месенджеру що встановлено на смартфоні, соціальної мережі яку він використовує, електронної пошти тощо.

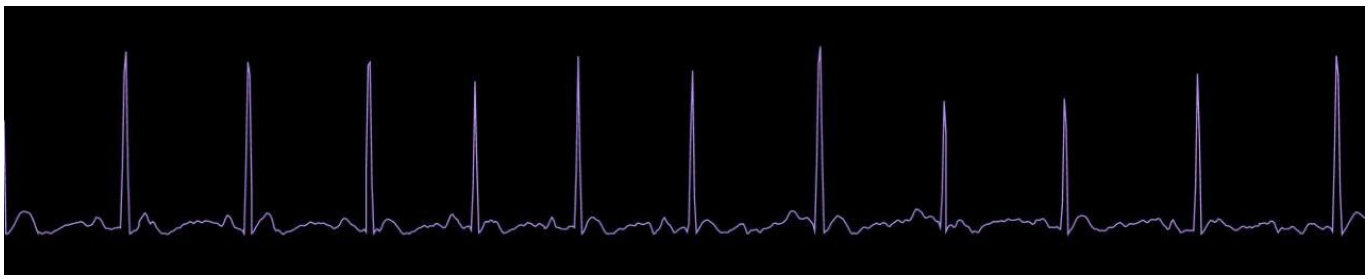


Рисунок 5.10 — Зображення відправленого результату ЕКГ

Дані оновлюються дуже часто – кожні пів секунди. Закриття мобільного додатку відбувається аналогічно усім іншим додаткам на смартфонах з операційною системою Android – помахом вгору. Для реалізації системи була обрана саме платформа Android через її популярність.

5.3 Висновки до розділу

П’ятий розділ містить інформацію стосовно системних вимог що необхідні задля роботи у додатку. Також у цьому розділі описано процес взаємодії користувача з мобільним додатком, наведено графічний приклад цього процесу.

ВИСНОВКИ

У ході виконання даної роботи була розроблена автоматизована система, що дозволяє отримувати ЕКГ сигнали портативним мобільним пристроям. Для розробки використовувалась мова програмування JavaScript та фреймворк React Native, який є необхідним для розробки додатку під операційну систему Android, а сам процес передачі здійснюється за допомогою технології Bluetooth. Відображення електрокардіограми у додатку здійснюється за допомогою обширної бібліотеки d3.js. Тестування програмного продукту було проведено за допомогою фреймворків Jest та Enzyme.

Необхідність подібного додатку була доведена за допомогою дослідження проблематики розповсюдженості серцево-судинних захворювань та демонстрування того факту, що електрокардіографія є одною з основних типів діагностик для відповідного виду захворювань.

Задля створення конкурентоспроможного додатку було проаналізовано сучасні системи що також дозволяють користувачам переглядати їх електрокардіограми. Особливу увагу було приділено їх недолікам, які були усунуті у розробленій системі.

Таким чином, розроблений мобільний додаток дозволяє користувачам:

- отримувати електрокардіограму;
- передивлятися встановлені норми електрокардіограми;
- отримувати пульс;
- передивлятися встановлені норми пульсу за різним віком;
- отримувати інформацію стосовно заряду приладу, який здійснюю вимір ЕКГ та пульсу;
- отримати інформацію стосовно додатку;
- ділитися електрокардіограмою за допомогою відправки зображення результату у месенджерах або на електрону пошту.

На разі додаток є доступним для використання виключно власниками

смартфонів на платформі Android, оскільки дана операційна система є найпопулярнішою в Україні. Таким чином, були здобуті навички розробки мобільних додатків, передачі даних за допомогою технології Bluetooth Low Energy, відображення пульсу користувача, заряду приладу у відсотках, створення електрокардіограми за допомогою отриманих даних та відправка отриманого результату за допомогою будь-якого доступного смартфону придатного для комунікації додатку. Тим більше, було отримано навички тестування програмного продукту за допомогою фреймворків Jest та Enzyme оскільки вчасне тестування та знайдення помилок при розробці є надзвичайно важливим для будь-якої системи що пов’язана з медициною.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Войтенко В.П. Україна в європейському контексті: кластерна модель смертності від головних причин / В.П. Войтенко, А.В. Писарук, Н.М. Кошель // Пробл. старения и долголетия. – 2014. – №1. – С.85–95.
2. Vagero D. The East-West Health Divide in Europe: Growing and Shifting Eastwards / D. Vagero // European Review. – 2010. – Vol. 18, № 1. – P. 23–34.
3. World Health Organisation. The Future Of CVD [Електронний ресурс] / World Health Organisation – Режим доступу до ресурсу: https://www.who.int/cardiovascular_diseases/en/cvd_atlas_25_future.pdf?ua=1 .
4. World Health Organisation. Metrics: Disability-Adjusted Life Year (DALY) [Електронний ресурс] / World Health Organisation – Режим доступу до ресурсу: https://www.who.int/healthinfo/global_burden_disease/metrics_daly/en/ .
5. Institute for Health Metrics and Evaluation. Глобальное бремя болезней : Порождение доказательств, направление политики [Електронний ресурс] / Institute for Health Metrics and Evaluation. – 2013. – Режим доступу до ресурсу: http://documents.worldbank.org/curated/en/785071468029977445/pdf/808480PUB0RUSS0Box0379820B00PUBLIC0.pdf .
6. ЗАБОЛЕВАНИЯ СЕРДЕЧНО-СОСУДИСТОЙ СИСТЕМЫ: ВИДЫ И ОСОБЕННОСТИ [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://samozdrav.ru/blog/zabolevaniya-serdechno-sosudistoy-sistemy/ .
7. Инсульт. Симптомы, виды, методы восстановления [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: http://kogalym-lpu.ru/informatsiya/rekomendatsii-vrachey/21144/ .
8. Heneghan C. A List of Cardiovascular Diseases: The 5 Most Common [Електронний ресурс] / Carolyn Heneghan. – 2018. – Режим доступу до ресурсу: https://www.dignityhealth.org/articles/a-list-of-cardiovascular-diseases-the-5-most-common .
9. Диагностика сердечно сосудистых заболеваний [Електронний ресурс] – Режим доступу до ресурсу: https://chekhovsc.ru/diagnostika-serdechno-sosudistykh-zabolevanij .
10. Goldberger A. Clinical Electrocardiography: A Simplified Approach / A. Goldberger, Z.

Goldberger, A. Shvilkin., 2012. – 352 c. – (8th Edition).

11. Чайковский И. А. Анализ кардиограммы в одном, шести и двенадцати отведениях с точки зрения информационной ценности: электрокардиографический каскад [Електронний ресурс] / И. А. Чайковский. – 2013. – Режим доступу до ресурсу: http://www.uacm.kharkov.ua/download/2013_10/48-58_Chaikovsky_10_sc_P.pdf .
12. The Selvester QRS Score is More Accurate than Q Waves and Fragmented QRS Complexes Using the Mason-Likar Configuration in Estimating Infarct Volume in Patients with Ischemic Cardiomyopathy [Електронний ресурс] / [M. Carey, A. Luisi, S. Baldwa та ін.]. – 2010. – Режим доступу до ресурсу: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2902677/ .
13. Jefferis N. Holter: A serendipitous life: An essay in biography / Norman Jefferis., 2008.
14. Freescale Semiconductor. Freescale Solutions for Electrocardiograph and Heart Rate Monitor Applications [Електронний ресурс] / Freescale Semiconductor. – 2013. – Режим доступу до ресурсу: https://www.nxp.com/docs/en/application-note/AN4323.pdf .
15. Сапитон М. СТАТИСТИКА ПОПУЛЯРНОСТИ СМАРТФОНОВ В УКРАИНЕ И ПОЧЕМУ ЭТО ВАЖНО [Електронний ресурс] / Михаил Сапитон. – 2016. – Режим доступу до ресурсу: https://uip.me/2016/07/smartphones-in-ukraine/ .
16. Айкожаев Н. М. АНАЛИЗ СРЕДЫ РАЗРАБОТКИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ ANDROID STUDIO [Електронний ресурс] / Н. М. Айкожаев, Н. А. Есеналиева. – 2017. – Режим доступу до ресурсу: https://sibac.info/archive/technic/2(49).pdf .
17. Камшилин А. Bluetooth: технология и ее применение [Електронний ресурс] / Артем Камшилин – Режим доступу до ресурсу: https://www.ixbt.com/mobile/review/bluetooth-2.shtml .
18. Qupta N. Inside Bluetooth Low Energy / Naresh Qupta. – London: Artech House, 2016. – 458 с. – (Second Edition).
19. Elliott E. Programming JavaScript Applications: Robust Web Architecture With Node, Html5, And Modern Js Libraries / Eric Elliott., 2013. – 253 с. – (First Edition).
20. Meeks E. D3.js in Action: Data visualization with JavaScript / Elijah Meeks., 2017. – 375 с. – (Second Edition).
21. Verou L. CSS Secrets: Better Solutions to Everyday Web Design Problems / Lea Verou., 2015. – 354 с. – (Third Edition).

Додаток 1

Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям

Специфікація

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62209_20Б

Аркушів 2

2020

Позначення	Найменування	Відмітки
Документація		
УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС _ТМ62209_20Б 81-1	Записка.docx	Пояснювальна записка
Компоненти		
УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС _ТМ62209_20Б 12-1	App.js	Основний компонент
УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС _ТМ62209_20Б 12-2	test.component.js	Компонент тестування
УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС _ТМ62209_20Б 13-1	Опис.docx	Опис програмного модулю

Додаток 2

Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям

Основний компонент

Текст програмного модулю

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62209_20Б 12-1

Аркушів 9

2020

```
//Імпорт необхідних компонентів зі сторонніх бібліотек
import React, {useEffect, useState, useCallback, useRef, Component} from 'react';
import {StyleSheet, Text, View, ScrollView, TouchableOpacity, Image} from 'react-
native';
import ReactDOM from 'react-dom';
import {BleManager} from "react-native-ble-plx";
import {Grid, LineChart} from 'react-native-svg-charts';
import * as d3Scale from 'd3-scale';

import ViewShot, { captureRef } from "react-native-view-shot";
import Share from 'react-native-share';
import RNFetchBlob from 'react-native-fetch-blob';

import Adapter from 'enzyme-adapter-react-16';
import Enzyme from 'enzyme';
Enzyme.configure({ adapter: new Adapter() });

//Алфавіт для шифрування
const chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';

const Base64 = {
    //Функція кодування строки в формат base64
    btoa: (input = '') => {
        let str = input;
        let output = '';

        for (let block = 0, charCode, i = 0, map = chars;
             str.charAt(i | 0) || (map = '=', i % 1);
             output += map.charAt(63 & block >> 8 - i % 1 * 8)) {

            charCode = str.charCodeAt(i += 3 / 4);

            if (charCode > 0xFF) {
                throw new Error("помилка у 'btoa': строка для зашифрування містить букви не
тільки латинського алфавіту");
            }

            block = block << 8 | charCode;
        }

        return output;
    },
    //Функція декодування строки з формату base64
    atob: (input = '') => {let str = input.replace(/=+$/, '');
                                    let output = '';

                                    if (str.length % 4 == 1) {
                                        throw new Error("помилка у 'atob': строка для
розшифрування була невірно зашифрована");
                                    }
                                    for (let bc = 0, bs = 0, buffer, i = 0;
                                         buffer = str.charAt(i++);

                                         ~buffer && (bs = bc % 4 ? bs * 64 + buffer :
buffer,

                                         bc++ % 4) ? output += String.fromCharCode(255
& bs >> (-2 * bc & 6)) : 0
                                    ) {
                                        buffer = chars.indexOf(buffer);
                                    }

                                    return output;

    }
};
```

```
function _base64ToArrayBuffer(base64) {
    let binary_string = Base64.atob(base64);
    let len = binary_string.length;
    let bytes = new Uint8Array(len);
    for (let i = 0; i < len; i++) {
        bytes[i] = binary_string.charCodeAt(i);
    }
    return bytes;
}

const bleManager = new BleManager();
const pmdID = 'fb005c80-02e7-f387-1cad-8acd2d8df0c8';
const pmdControlChar = 'fb005c81-02e7-f387-1cad-8acd2d8df0c8';
const pmdMTUchar = 'fb005c82-02e7-f387-1cad-8acd2d8df0c8';
const bataryId = '180f';

const parseECGData = (data, time = 0)  => {
  //Відкидання зайвої додаткової інформації
  let items = data.slice(10);
  //Групування байтів відповідаючих за одне значення
  let newArr = [];
  for (let i = 0; i < items.length; i += 3) {
    let arr;
    if (typeof items[i + 2] === 'number') {
      arr = [items[i], items[i + 1], items[i + 2]];
      newArr.push(arr);
    }
  }
  //Формування цілісного значення з окремих байтів
  return newArr.map((item) => {
    const newItem = item.map((subItem) => {
      //Приведення в шістнадцяткову систему числення
      const num = subItem.toString('16');
      //Доповненя байтів зі значенням менше 16 для того щоб кожне число(байт) складалося з двух
цифр
      return num.length === 1 ? `0${num}` : num;
    });
    //Приведення в десяткову систему числення
    return parseInt(newItem.reverse().join(''), 16)
    //Фільтрування аномально високих значень викликаних похибкою під час замірів
  }).filter( data => data < 1000).map(item => {
      //Формування пари показник ЕКГ <---> час заміру
      const newItem = {
          item,
          time
      };
      time+=0.008;
      time = +time.toFixed(3);
      return newItem;
  });
}

// Основний робочий компонент додатку
export default function App() {

    // Синтаксис React Hook - [state, function] = useState(init)
    // дозволяє задати стан state та функцію зміни стану function
    // знвчення init є початковим станом
    let [accLevel, setAccLevel] = useState();
    let [device, setDevice] = useState();
    let [ecgData, setEcgData] = useState([]);
    let [mock, mockSet] = useState(null);
    let [heartRate, setHeartRate] = useState(0);
    let [appInfoDisplay, setAppInfoDisplay] = useState(false);
    let [pulseInfoDisplay, setPulseInfoDisplay] = useState(false);
    let [ECGInfoDisplay, setECGInfoDisplay] = useState(false);
    let [photos, setPhotos] = useState([]);
```

```
    let [index, setIndex] = useState(null);

    // React Hook useRef дозволяє зберігати послання на компонент для подальшої взаємодії
    let imgRef = useRef();

    const pulsePopupText = 'Відповідно до Всесвітньої Організації Здоров`я, наразі встановлені такі норми пульсу: \n' +
                                    'Діти на першому місяці життя - середній показник 140 ударів на хвилину, мінімальний 110, а максимальний 170. \n' +
                                    'Діти до року - середній показник 132, мінімальний 102, а максимальний 162  ударів на хвилину. \n' +
                                    '4-6 років - середній показник 106, мінімальний 86, максимальний 126  ударів на хвилину. \n' +
                                    '6-7 років - середній показникі 98, мінімальний 78, максимальний 118  ударів на хвилину. \n' +
                                    '8-10 років - середній показник 88, мінімальний 68, максимальний 108  ударів на хвилину. \n' +
                                    '10-12 років - середній показник 80, мінімальний 60, максимальний 100  ударів на хвилину. \n' +
                                    '12-15 років - середній показник 75, мінімальний 55, максимальний 95 ударів на хвилину. \n' +
                                    '15-50 років - середній показник 70, мінімальний 60, а максимальний 80 ударів на хвилину. \n' +
                                    '50-60 років - середній показник 74, 64 – нижня межа норми, 84 – верхня. \n' +
                                    '60+ років - нормальним пульсом вважається 79 ударів, 69 мінімально допустимий і 89 максимально допустиме значення.';

    const pulseImg = require('./assets/doctor.png');

    const ECGPopupText = 'Нормою для дорослої людини є такий рівень характеристичних показників ЕКГ: \n' +
                                'Інтервал RR: 0,6-1,2 секунд \n' +
                                'Інтервал PR: 120-200 мілісекунд \n' +
                                'Інтервал ST: 320 мілісекунд \n' +
                                'Інтервал QT: 420 мілісекунд або менше \n' +
                                'P-зубець – 80 мілісекунд \n' +
                                'T-зубець – 160 мілісекунд \n' +
                                'Для дітей ці норми будут відрізняться. \n' +
                                'Якщо ви спостерігаєте суттєве відхилення від приведеного малюнка та показників варто відвідати лікаря кардіолога. \n' +
                                'Також варто звернути увагу на регулярність ритму, не постійний ритм є вагомою причиною звернутися до лікаря.';

    const ECGImg = require('./assets/ecg.png');

    const appPopupText = 'Даний додаток розроблений задля моніторингу здоров`я серцево-судинної системи за допомогою пульсу та електрокардіограми.\n' +
                                'Будь ласка, пам`ятайте що встановлення дігнозу можливе тільки лікарем за допомогою додаткових досліджень. \n' +
                                'Самолікування може бути шкідливим для вашого здоров`я.';

    const appInfoImg = require('./assets/appinfo.png');

    // React Hook useEffect використовується для виконнання коду одразу після рендеру компоненту
    useEffect(() => {
        if (!device) {
          bleManager.onStateChange((newState) => {
            if (newState === 'PoweredOn') {
              //Ініціалізація пошуку пристрою
              bleManager.startDeviceScan([], {allowDuplicates: true}, async (err, device) => {
                if (device) {
                  if (device && device.name && device.name.startsWith('Polar')) {
```

```
                    //Припинення пошуку пристрою та підключення до знайденого
                    bleManager.stopDeviceScan();
                    await bleManager.connectToDevice(device.id, {
                      requestMTU: 512
                    });
                    setDevice(device);
                    //Запит списку характеристик пристрою
                    const scanned = await
device.discoverAllServicesAndCharacteristics();
                    const data = await scanned.services();
                    for (let key in data) {
                      if (data[key].uuid.includes('180f')) {
                        //Робота з сервісом батареї по ідентифікатору сервісу 180f
                        const chars = await data[key].characteristics();
                        console.log('chars for battery', chars);
                        //Запит списку характеристик сервісу, вибір характеристики заряду батареї яка
нас цікавить
                        const bat = await
device.readCharacteristicForService(data[key].uuid, chars[0].uuid);
                        setAccLevel(_base64ToArrayBuffer(bat.value)[0]);
                      }
                        if (data[key].uuid.includes('180d')) {
                            //Робота з сервісом пульсу по ідентифікатору сервісу 180d
                            const chars = await data[key].characteristics();
                            console.log('chars for heart rate', chars[0].uuid);
                            //Запит списку характеристик сервісу, вибір характеристики ударів за
хвилину яка нас цікавить
                            await
device.monitorCharacteristicForService(data[key].uuid, chars[0].uuid, (err, data) => {
                                setHeartRate(_base64ToArrayBuffer(data.value)[1]);
                            });
                        }
                    }
                    device.monitorCharacteristicForService(pmdID, pmdMTUchar, (err,
data) => {
                      //Робота з характеристикою ЕКГ
                      if (data && data.value && (data.value).length > 20) {
                        const newArr = parseECGData(_base64ToArrayBuffer(data.value),
ecgData && ecgData[ecgData.length-1] && ecgData[ecgData.length-1].time || 0);
                        ecgData.push(...newArr);
                        mockSet(newArr);
                      }
                    });
                    const res = await
device.writeCharacteristicWithResponseForService(pmdID, pmdControlChar,
'AgAAAYIAAQEOAA==');
                     console.log('stream response', _base64ToArrayBuffer(res.value));
                  }
                }
              })
            }
          }, true)
        }
    }, []);
    const dataSet = ecgData.slice(-739);
        return (
            <View style={styles.wrapper}
contentContainerStyle={common.containerCenter}>
                <Text style={styles.battery}>
                    Заряд: {accLevel}%
                </Text>
                <View contentContainerStyle={common.containerCenter}>
                    <Text style={styles.pulse}
contentContainerStyle={common.containerCenter}>
                        {heartRate}
                    </Text>
                </View>
```

```
                <View style={styles.btnWrap}
contentContainerStyle={common.containerCenter}>
                    <Button text={'Опис додатку'} clickFunc={setAppInfoDisplay}/>
                    <Button text={'Норми пульсу'} clickFunc={setPulseInfoDisplay}/>
                    <Button text={'Норми ЕКГ'} clickFunc={setECGInfoDisplay}/>
                    <SaveBtn text={'Поділитися ЕКГ'} imgref={imgRef}/>
                </View>

                // Компонент ScrollView для отримання можливості гортати графік
                <ScrollView horizontal={true}
contentContainerStyle={common.containerEnd}>
                    // Компонент ViewShot для зазначення області знімку
                    <ViewShot collapsable={false} ref={imgRef}
snapshotContentContainer={true}>
                        <LineChart
                        style={{height: 200, minWidth: 100 * dataSet.length/73}}
                        contentContainerStyle={{justifyContent: 'flex-end'}}
                        data={dataSet}
                        keys={['item', 'time']}
                        colors={['#8800cc', '#aa00ff']}
                        contentInset={{top: 30, bottom: 30}}
                        svg={{stroke: '#bd93f9'}}
                        xAccessor={(item) => item.item.time}
                        yAccessor={(item) => item.item.item}
                        numberOfTicks={73}
                        >
                        <Grid/>
                        </LineChart>
                    </ViewShot>
                </ScrollView>

                // Компоненти Popup розміщуються в кінці основного компоненту
                // та викликаються при натисканні відповідних кнопок
                <Popup blockDisplay={appInfoDisplay}
blockDisplayFunc={setAppInfoDisplay} popupText={appPopupText} popupImg={appInfoImg} />
                <Popup blockDisplay={pulseInfoDisplay}
blockDisplayFunc={setPulseInfoDisplay} popupText={pulsePopupText} popupImg={pulseImg}
/>
                <Popup blockDisplay={ECGInfoDisplay}
blockDisplayFunc={setECGInfoDisplay} popupText={ECGPopupText} popupImg={ECGImg} />

            </View>
        );
}

//Компонент кнопки, в якому ми вказуємо текст кнопки та функцію
//виконання якої відбувається при натисканні
export const Button = ({text, clickFunc}) => {
    return (
        <TouchableOpacity
        style={styles.touchable}
        activeOpacity={0.9}
        //Створення зв`язку між компонентами Button <---> Popup
        //за допомогою функції керування станом
        onPress={() => {clickFunc(true)}}>
            <Text
             style={styles.btn}
             contentContainerStyle={common.containerCenter}>
                {text}
            </Text>
        </TouchableOpacity>
    );
}

//Компонент спеціальної кнопки для ініціалізації збереження ЕКГ знімку
export const SaveBtn = ({text, imgref}) => {
    return (
```

```
        <TouchableOpacity style={styles.touchable} activeOpacity={0.9} onPress={() =>
{
            //Створення зображення з контрольної області в межах додатку
            captureRef(imgref, {
              format: "jpg",
              quality: 0.8
            }).then(
              //uri - посилання на створене зображення
              uri => {
                const image = uri;
                //Отримання представлення зображення у вигляді base64 строки
                RNFetchBlob.fs.readFile(image, 'base64')
                .then((data) => {
                  //Формування коректного об'єкту данних
                  let shareOptions = {
                    title: "ЕКГ",
                    message: "Збережене ЕКГ",
                    url: `${data}`,
                    subject: "Збережене ЕКГ"
                  };
                  //Використання стандартної для пристрою операції "поділитись зображенням/файлом"
                  Share.open(shareOptions)
                    .then((res) => console.log('res:', res))
                    .catch(err => console.log('err', err))
                });
              },
              error => console.error("Щось пішло не так під час процесу відправки ЕКГ",
error)
            );
        }}>
            <Text style={styles.btn} contentContainerStyle={common.containerCenter}>
                {text}
            </Text>
        </TouchableOpacity>
    );
}

//Компонент для демонстрації додаткової інформації за потребою
export const Popup = ({blockDisplay,blockDisplayFunc,popupText,popupImg}) => {
    //Використання стану React Hook - useState для відображення компоненту
    let displayProp = blockDisplay ? 'flex' : 'none';
    let positionProp = blockDisplay ? 'absolute' : 'relative';

    return (
        <ScrollView style={{display: displayProp, position: positionProp, zIndex: 20,
top: 0, left: 0, width: 410, height: 800, backgroundColor: '#282a36'}}>
            //Передача керуючої станом функції для утворення зв'язку
            //між компонентами Popup <---> CloseBtn
            <CloseBtn text={'close'} clickFunc={blockDisplayFunc}/>
            <Image source={popupImg} style={{width: 420, marginBottom: 10, marginTop:
20}} />
            <Text style={styles.popup}>{popupText}</Text>
        </ScrollView>
    );
}

//Спеціальний підвид компоненту кнопки для використання в контексті компоненту Popup
//для повернення до головного єкрану
export const CloseBtn = ({text,clickFunc}) => {
    const popupCloseImg = require('./assets/close.png');
    return(
        <TouchableOpacity
        style={styles.closeBtn}
        activeOpacity={0.9}
        onPress={() => {clickFunc(false)}}>
            <Image source={popupCloseImg} style={{width: 40, height: 40}} />
        </TouchableOpacity>
```

```
    );
}

//Згруповані стилі дозволяють відділити їх від структурної реалізації
//покращують читаємість коду та дозволяють перевикористання стилів
const common = new StyleSheet.create({
    containerCenter: {
        justifyContent: 'center',
        alignItems: 'center'
    },
    containerEnd: {
        justifyContent: 'flex-end',
        alignItems: 'flex-end'
    },
});
const styles = new StyleSheet.create({
    wrapper: {
        flex: 1,
        backgroundColor: '#282a36'
    },
    btn: {
        fontFamily: 'Open Sans',
        fontSize: 20,
        color: '#bd93f9',
        textAlign: 'center',
        lineHeight: 40,
        width: 200,
        height: 40,
        borderWidth: 3,
        borderRadius: 100,
        borderColor: '#bd93f9',
        backgroundColor: '#f8f8f2'
    },
    touchable: {
        width: 200,
        height: 40,
        marginLeft: 'auto',
        marginRight: 'auto',
        marginTop: 30
    },
    popup: {
            fontFamily: 'Open Sans',
            flex: 1,
            fontSize: 15,
            color: '#50fa7b',
            margin: 7,
            borderRadius: 12,
        },
    battery: {
        fontFamily: 'Open Sans',
        flex: 1,
        fontSize: 20,
        color: '#50fa7b',
        paddingLeft: 10,
        paddingTop: 10,
        borderRadius: 100,
    },
    pulse: {
        fontFamily: 'Open Sans',
        fontSize: 48,
        color: '#ff79c6',
        textAlign: 'center',
        lineHeight: 95,
        width: 100,
        height: 100,
        borderWidth: 5,
        borderRadius: 100,
        borderColor: '#ff79c6',
```

```
        backgroundColor: '#f8f8f2',
        marginLeft: 'auto',
        marginRight: 'auto',
        marginBottom: 60,
    },
    btnWrap: {
        flex: 1,
        minHeight: 220
    },
    closeBtn: {
        width: 40,
        height: 40,
        marginLeft: 'auto',
        marginBottom: 10,
        marginTop: 20,
        marginRight: 20
    }
});
```

Додаток З

Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям

Опис програмного модулю

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62209_20Б 13-1

Аркушів 8

2020

АНОТАЦІЯ

У додатку здійснюється отримування та відображення даних електрокардіограми за допомогою мобільного пристрою та відповідного приладу що надає потрібну інформацію для відображення ЕКГ.

У системі виконуються наступні функції:

— підключення до приладу за допомогою Bluetooth;

— відображення ЕКГ та пульсу;

— отримання інформації стосовно додатку, норм електрокардіограми та пульсу;

— відправка отриманих результатів ЕКГ.

Додаток був розроблений за допомогою мови програмування JavaScript та фреймворку React Native.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ 67

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ.....68

ОПИС ЛОГІЧНОЇ СТРУКТУРИ 69

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ.....70

ВИКЛИК І ЗАВАНТАЖЕННЯ.....71

ЗАГАЛЬНІ ВІДОМОСТІ

Розроблений додаток здійснює процес отримування та відображення сигналів ЕКГ та пульсу, відправку отриманих результатів електрокардіографії та надання інформації стосовно ЕКГ та пульсу.

Модулі системи описані в ДОДАТКУ З.

Завантаження та використання системи можливо на смартфонах з операційною системою Android з версію 8.0 та вище, технологією Bluetooth. Для використання додатку необхідно мати прилад з відповідними сенсорами для отримання ЕКГ та пульсу (наприклад, Polar H10).

Використана мова програмування — JavaScript.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблена автоматизована система призначена для стеження за здоров’ям серцево-судинної системи користувача за допомогою електрокардіографії та пульсу .

Даний додаток може використовуватися будь-якою людиною що зацікавлена у стеженні за здоров’ям серцево-судинної системи у будь-якому місці оскільки електрокардіографія немає ніяких протипоказань та для використання потрібен тільки мобільний пристрій та відповідний прилад з сенсорами для отримування ЕКГ та пульсу.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Автоматизована система являє собою додаток з інтуїтивно зрозумілим та простим інтерфейсом, що дозволяє без будь-яких перешкод отримувати користувачу результати ЕКГ та пульс, передивлятися відповідну інформацію стосовно додатку, норм що встановлені ВОЗ та відправляти отриману електрокардіограму за допомогою месенджера, соціальної мережі або електронної пошти.

Для використання додатку потрібно лише мати смартфон з операційною системою Android та прилад для отримування ЕКГ сигналів. Використання системи можливе після завантаження відповідного APK-файлу та встановлення додатку. Для успішного підключення до приладу необхідне включення функції Bluetooth. Після підключення користувач має змогу отримувати пульс та електрокардіограми, відправляти результати ЕКГ за допомогою мережі інтернет.

ВИКОРИСТОВУВАНІ ТЕХІЧНІ ЗАСОБИ

Для розробки автоматизованої системи використовувалося середовище розробки Android Studio, мова програмування JavaScript та фреймворк React Native, тестування додатку відбулося за допомогою бібліотек Jest та Enzyme.

Додаток працює на смартфонах з операційною системою Android з версією 8.0 та вище. Для використання не потрібно встановлення будь-яких додаткових компонентів, тільки встановлення самого додатку.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для використання необхідно встановити додаток, ввімкнути функцію Bluetooth та надягнути прилад для отримання сигналів ЕКГ та пульсу. Отримання інформації стосовно додатку, норм ЕКГ та пульсу що встановлені ВОЗ не потребує підключення до приладу. Після підключення до приладу відбувається поступове відображення електрокардіограми та пульсу.

Відправка результатів ЕКГ потребує підключення до мережі інтернет та робить можливим відправку результатів за допомогою будь-якого зручного для користувача методу: електронна пошта, месенджер або соціальна мережа.

Додаток 4

Автоматизована система передачі сигналів ЕКГ портативним мобільним пристроям

Теза доповіді Міжнародної науково-практичної конференції

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТМ62209_20Б

Аркушів 2

2020

Серцево-судинні захворювання (ССЗ) – це група захворювань серця та кровоносних судин, які є основною причиною смерті в усьому світі – щорічно від ССЗ помирає більше людей, ніж від будь-якої іншої хвороби. Згідно звіту Інститут геронтології імені Д.Ф Чеботарьова НАМН України, у структурі смертності щороку в Україні понад 68% осіб помирають через серцево-судинні хвороби. Це – близько 420-430 тисяч людей. Таким чином, за два роки Україна втрачає таку кількість населення, яка проживає у таких містах як Львів або Дніпро. Тим більше, щороку констатується від 40 до 50 тисяч випадків інфарктів в Україні, що робить проблему ССЗ дуже гострою в нашій країні. Існує програма створення реперфузійних центрів, але самі по собі вони не вирішують проблему бо найважливішою є профілактика цього типу захворювань та слідкування за своїм здоров'ям.

Таким чином, програмний продукт який відображає ЕКГ сигнали на мобільному пристрої та повідомляє користувача у разі перевищення встановленої норми може зберегти велику кількість людських життів та попередити незворотний негативний вплив ССЗ на здоров'я. Цей «продукт» призначений для пересічних громадян країни та не потребує великих витрат оскільки все, що потрібно для слідкування за роботою серця у цьому разі – це мобільний пристрій, на який потрібно встановити розроблений додаток та розміщений на тілі нагрудний пульсометр. Наприклад, Polar H10 є одним з найточніших та найзручніших пульсометрів на ринку у даний час [1].

Для реалізації додатку була обрана мова розмітки HTML та CSS, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних. В якості СКБД при вирішенні даної задачі використовується Microsoft SQL Server, для розробки фізичного представлення бази даних використовується MySQL Workbench. Також в роботі використовується скриптова мова JavaScript та бібліотека з відкритим кодом jQuery. Мобільність додатку була забезпечена завдяки відомому фреймворку для створення мобільних прикладних програм – Apache Cordova [2].

Розроблену програму можна використовувати для спостереження за електричною активністю серця дешевим, ефективним та простим у користуванні способом з мобільного телефону або планшета користувача. Також можна використовувати програму під час вправ або для спостереження за станом серця, що дійсно робить її придатною для використання підчас різноманітних фізичних навантажень та слідкування за здоров'ям у повсякденному житті. Тим більше, за допомогою додатку користувач може вислати результати електрокардіографії лікарю та проконсультуватися у домашніх умовах. Таким чином, процедура слідкування за своїм здоров'ям стає незвично простою та зручною для кожної людини.

Перелік посилань;

1. Хемптон Д. ЕКГ у практиці / Джон Р. Хемптон. – Київ, 2018. – 560 с.

2. HEALTH CARE MONITORING FOR THE CVD DETECTION USING SOFT COMPUTING

TECHNIQUES [Електронний ресурс] // International Journal in Foundations of Computer

Science & Technology (IJFCST). – 202. – Режим доступу до ресурсу:

https://www.academia.edu/12680638/HEALTH_CARE_MONITORING_FOR_CVD_DETECTION_USING_SO

FT_COMPUTING_TECHNIQUES.